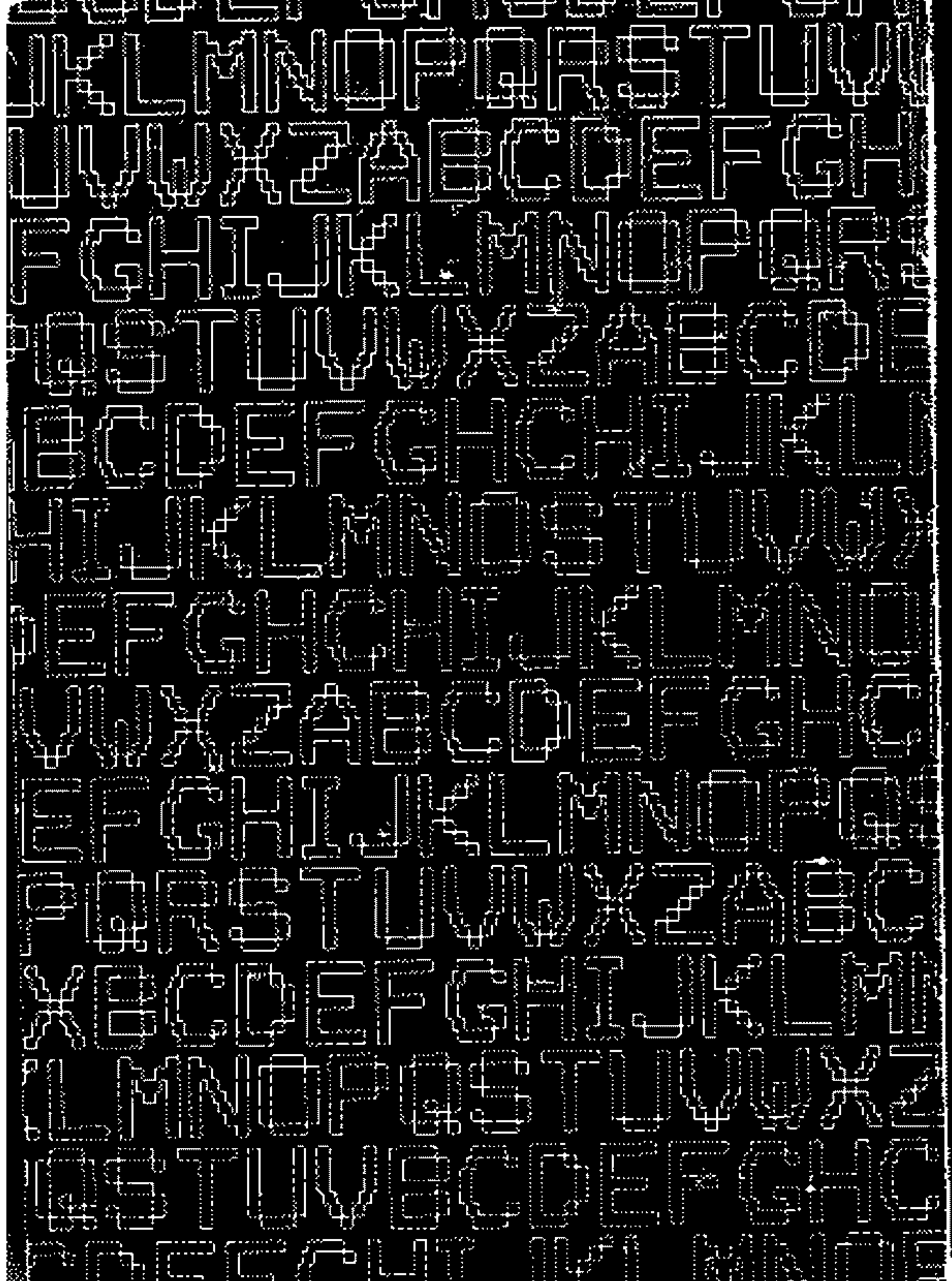


ZÁKLADY ČINNOSTI IQ 151

OBSAH

1. Úvod

Mikropočítač se nám může jevit jako personifikované zařízení, které nás slepě poslouchá, či s námi (někdy) tvrdohlavě neníní komunikovat. Je to hračka pro děti i rodiče, je to dobrý spolupracovník či společník v zaměstnání, ve škole i doma.

Mikropočítače radikálně změnily filosofii při užití výpočetní techniky. Velké počítače nám pomáhaly již dříve. Ač pracují mnohem rychleji, musel jejich uživatel za nimi cestovat, musel mít přidělený strojový čas atd. Převážné většině uživatelů však je jedno, je-li problém vyřešen za milisekundu či za minutu (řečeno obrazně). Proto vzrostl zájem o mikropočítače, které jsou relativně dostupné a jsou okamžitě přístupné uživateli.

Tato knižní publikace by chtěla uživatelům mikropočítačů, zvláště pak uživatelům školního mikropočítače IQ 151, prohloubit znalosti o konstrukci a činnosti mikropočítače. Je určena učitelům i žákům či i širší veřejnosti, která chce poznat jak vlastně mikropočítač pracuje. Populárním výkladem by měla dokázat, že činnost mikropočítače není něčím nepochopitelným, ale že je objasnitelná i uživatelům bez hlubších znalostí z elektroniky.

Šíří problematiky chce autor poukázat na některé mezery, které by si měl vážnější uživatel mikropočítače doplnit, aby mohl svůj mikropočítač kvalitativně lépe využít. První kapitola přibližuje situace, za kterých vznikl mikropočítač a mikroprocesor. Snaží se přiblížit čtenáři i výrobu a technologie integrovaných obvodů, rovněž jsou připomenuty některé základní pojmy. Druhá kapitola je věnovaná základním pojmům z technického řešení mikropočítače. Třetí kapitola pak programovému vybavení mikropočítače. Kapitolu čtvrtou uvítají jistě všichni uživatelé IQ 151, neboť zde naleznou konkrétní prakticky využitelné informace, mající vztah

k problematice druhé a třetí kapitoly.

N.p. Komenium vydalo k mikropočítači IQ 151 již tyto knižní publikace:

1. Kollert, E.: Programování počítače IQ 151 v jazyku BASIC, (1984)
2. Jedlička, Z., Feil, M.: BASIC pro začátečníky, (1985)
3. Feil, M.: MONITOR IQ 151, (1986)
4. Feil, M.: Strojový kód IQ 151, (1986)

Tato následující knižní publikace tedy již vychází ze znalostí v uvedených publikacích a často se na ně odvolává. Rovněž je nutné upozornit, že tato publikace si neklade za cíl býti elektrotechnickou encyklopedií od PN přechodu přes tranzistor, integrovaný obvod až po mikroprocesor. Jistě se vyskytnou pojmy, které nebudou všem čtenářům známé, či opačně, budou-li si vážní zájemci chtít některé znalosti důsledně osvojit, pak jsou odvoláni na literaturu (převážně československou). Některé kapitoly jsou napsány popisně a měly by čtenáři pouze poodhalit mikropočítačový rozhled či usnadnit orientaci v některých nových pojmech.

Čtvrtá kapitola je psána problémovým přístupem. Předpokládá, že si čtenář probranou či nadnesenou problematiku sám ověří na mikropočítači IQ 151. Tato kapitola je dosti náročná, je nutné ji studovat, nikoliv pouze přečíst. Ověřené a zvládnuté činnosti mikropočítače nechť si čtenář sám poznamená do jakési svojí „kuchařky" IQ 151. Na složitější problematiku je čtenář v textu upozorněn. Může se k ní při hlubším studiu později vrátit.

Pozn.: Velké písmeno K značí 1Ø24 . 1KB je tedy 1Ø24 bytů. Označení velké B místo byte je zvoleno, abychom nemuseli slovo byte skloňovat. Nuly jsou psány vesměs počítačově ... Ø .

Hexadecimální čísla jsou označena písmenem velké H ... např. 13H je hexadecimální číslo 13 .

Autor

1.1. Historie vzniku mikropočítače

Tuto kapitolu nemusí čtenář či uživatel mikropočítače studovat ba ani číst. Je jakýmsi všeobecně poznávacím vstupem (zčásti historickým, spíše však vědecko-„fantastickým") do problematiky mikropočítačů. Takováto „vyprávění" o počítači jsou v současné době předmětem mnoha článků či knih /6/, /8/, aj.

Vědní disciplína historie počítače ještě neexistuje. Avšak o prudkém rozvoji počítačové techniky by se dalo napsat mnohosvazkové dílo. Ne, nebojte se, nezačneme od vrubovek, sčotů a mechanických Pascalových sčítaček. Skočíme rovnou do dvacátého století.

První komu se podařilo sestrojit fungující počítací stroj byl Němec Konrád Zuse. Tento počítač nesl označení Z 1 a začal počítat v roce 1936. Byl elektromechanický s kolíčkovou pamětí na 16 čísel. Počítač byl nespolehlivý a proto se tentýž autor pustil do stavby počítače Z 2 (asi 200 relé) a posléze Z 3 (asi 2.600 relé). Přibližně ve stejné době v USA začal pracovat Dr. Howard Aiken na projektu automatického sekvenčně řízeného počítače (ASCC). Projekt financovala firma IBM, která se v počítačovém světě skloňuje snad ve všech pádech. Počítač byl dokončen v roce 1943 na Harvardské universitě a dostal název MARK I. Toto 15 metrů dlouhé zařízení mělo jako základní hnací jednotku elektromotor (příkonu 3,7 kW) s dlouhým hnacím hřídelem, která zprostředkovala pohon jednotlivých částí počítače. Program byl na děrné pásce, která měla 24 stop. Mark I dovedl sečíst dvě čísla za 0,3 s a vynásobil je za 6 s. Sinus daného úhlu vypočítal za jednu minutu.

Poté přišel na svět MARK II od téhož autora. Zařízení bylo již pouze reléové. Sečtení dvou čísel trvalo 0,125 s násobení asi 0,25 s. Celý počítač byl zkonstruován z 13.000 relé.

Reléové počítače vznikaly i v socialistických zemích. RVM 1 zkonstruovaný v SSSR ing. I.I. Bessovem v r. 1957 provedl až 20 násobení za sekundu. V ČSSR (v roce 1958) byl zkonstruován samočinný počítač SAPO (obsahoval 7.000 relé a 400 elektronek), případně v této době špičkové řešení československého počítače EPOS 1 a 2.

Do konstrukce počítačů vstupuje nový prvek - bistabilní klopný obvod zkonstruovaný ze dvou elektronek, který vytlačil relé. Během II. světové války sestrojil J. Prosper a J. Mouchly na Pensylvánské universitě v USA počítač ENIAC. Obsahoval 18.000 elektronek a 1.500 relé, měl příkon 200 kW a byl chlazen dvěma leteckými motory. ENIAC pracoval rychlostí 5.000 operací za sekundu. Tehdy kuriózní (v současné době u superpočítačů běžné) bylo též paralelní zpracování operací.

Kolem roku 1960 se začaly objevovat první tranzistorové počítače. Na trh byly dodány jednoduché logické obvody TTL. Zatímco velké počítače a minipočítače byly doménou velkých firem, mikropočítače přivedli na svět amatéři. V roce 1966 založil S.B. Cray spolek ACS (Amateur Computer Society). V jeho časopise ACS Newsletter v letech 1966-1968 byly popsány konstrukce amatérských domácích počítačů. Podle údajů amatérů trvala stavba až dva roky.

Prudký nástup mikropočítačů můžeme datovat rokem 1969. Japonská firma Busicom zadala firmě Intel zakázku na výrobu integrovaných obvodů pro kalkulátor s tiskárnou (dle /4/) či pro elektronické pokladny (dle /7/). A tak se objevil první 4 bitový mikroprocesorový systém, jehož základními součástkami byl mikroprocesor I 4ØØ4, paměť ROM 4ØØ1 (256 byte), paměť RAM 4ØØ2 (20 4bitových slov). V roce 1972 firmy Intel a Texas uvedly na trh nový mikroprocesor I 8ØØ8, což už byl 8bitový mikroprocesor. Vyžadoval kolem sebe rozsáhlý soubor podpůrných obvodů nižší

integrace. V roce 1973 přichází firma Intel s mikroprocesorem I 8Ø8Ø s mnohem pokrokovější elektronickou architekturou. První mikropočítače s těmito mikroprocesory měly podobu mikropočítačových stavebnic. Po první úspěšnější stavebnici Mark-8 popsal H. Edward Roberts stavebnici Altair 88ØØ (r. 1975). Malá firma MITS čítající 20 zaměstnanců byla zahrnuta hned poté objednávkami stavebnic. Do konce roku 1976 prodala asi 10.000 stavebnic. Sběrnice této stavebnice nazvaná S-100 se stala standardem i pro jiné počítače. Tato stavebnice měla na ovládacím panelu 24 páčkových přepínačů na adresování paměti, zadávání dat (16+8) a 36 LED diod (16 indikuje stav adresové sběrnice, 8 stav sběrnice dat, 12 stav systému). Logika umožňovala spustit mikroprocesor na dané adrese, krokovat program, modifikovat data zadaná páčkovými přepínači, vynulovat data aj. (Zde si neodpustím poznámku, že 10.000 amerických amatérů začínalo s „páčkovými mikropočítači" a přesto je v samých začátcích neodložili, ale naopak dále zdokonalovali. Naše veřejnost dostala do vínku mikropočítače typu PMD-85 či IQ 151 aj., které jsou proti Altair 88ØØ úplně luxusní mikropočítače.

V roce 1977 se začíná přecházet od mikropočítačových stavebnic k osobním počítačům, kterými se rozumí stolní provedení včetně alfanumerické klávesnice, zobrazovací jednotky a využití vyššího programovacího jazyka. Příkladem je osobní mikropočítač Apple II či PET 2ØØ1 (oba BASIC 8KB, RAM 8KB), či TRS 8Ø. Základní konfigurace stála asi 2.000 dolarů.

Velkým konkurentem pro výše uvedené firmy se stal mikropočítač ZX-8Ø z Velké Británie. V roce 1980 ovládl trh. Miniaturizace slavila úspěch, rovněž tak minimalizace technického řešení (na úkor programového vybavení), nejvíce však cena 100 liber. Sto padesát tisíc kusů se prodalo během jednoho roku. Následovala

novinka ZX-81, která byla proti všem předpokladům levnější (!) . Počítač tvořily pouze čtyři integrované obvody. Úspěch byl obrovský, zanedlouho výroba překročila 100.000 kusů měsíčně. V současné době je na světě asi 3 milióny těchto počítačů (v ČSSR asi 20.000 kusů). V roce 1982 nabídla firma Sinclair novinku - ZX Spectrum, který je charakterizován velkou pamětí, osmibarevnou grafikou a dalšími vylepšeními. Obchodní úspěch byl opět veliký (v ČSSR je asi 40.000 kusů). V této době existují ve světě stovky mikropočítačů v kategorii domácí až profesionální. Orientovat se v této nabídce není jednoduché. Vítězí buď ekonomická stimulace či profesionální kvalita, avšak též i dostupnost a kvalita programového vybavení. Sharp, Commodore, IBM, Epson, Hewlett Packard, DEC, Apple, Sinclair jsou firmy, které se v letech 1983 až 1986 drží mezi nejúspěšnějšími výrobci mikropočítačů. Koordinace mezi výrobci prakticky neexistuje. V oblasti větší výpočetní techniky představuje jistý standard IBM a totéž platí i o osobních počítačích velikosti IBM PC, kde si žádný výrobce nedovolí vyrábět počítač, který by nebyl s IBM PC kompatibilní, neboť naděje na prodejní úspěch by byly malé. Vedle této autoritativní kompatibility přišli Japonci se zavedením standardu MSX viz /1/, /2/. Standard MSX zahrnuje jak mikroprocesor (Z-8Ø), tak minimální konfiguraci RAM, ROM, videoprocesor, zvukový generátor, formát záznamu na kazetový magnetofon, ba i programovací jazyk Mikrosoft BASIC. Na standard MSX sice přistoupily i některé významnější firmy, avšak zda se MSX prosadí, ukáže až budoucnost.

Vrátíme se trochu zpět k technickému řešení mikropočítačů. V roce 1973 jsme konstatovali nástup mikroprocesoru 8Ø8Ø. Tento mikroprocesor je 8-bitový s možností přímé adresace 64KB, počet základních instrukcí 78. Další typ, který minimalizuje podpůrné obvody mikroprocesoru, byl 8Ø85 (instrukční soubor i možnosti

jsou o málo lepší než u 8080). Firma Zilog (USA) v roce 1976 uvedla revolučnější mikroprocesor Z-80. Vychází z instrukčního souboru 8080, se kterým je plně kompatibilní. Navíc umožňuje dalších 80 nových instrukcí, má větší možnost adresace, jedno napájení a sdružuje některé podpůrné obvody.

Rokem 1978 začíná nástup nové generace v mikroprocesorové technice reprezentované 16bitovými mikroprocesory. V roce 1978 to byl mikroprocesor 8086. Délka slova 16bitů, kapacita adresace 1MB, počet základních instrukcí je 97. Tento typ mikroprocesoru spolu s podpůrnými aritmetickými a vetupně výstupními mikroprocesory (tzv. koprocesory) se stává „klasickým" řešením střední třídy mikropočítačů.

Vývoj mikroprocesorů však jde dál. Firma Motorola (USA) konstruuje 32-bitový mikroprocesor MC 68000, kapacita adresace 16MB, počet základních instrukcí je 63, avšak jako základní instrukce je považováno i násobení dvou 16-bitových čísel.

Kompletní mikropočítače každým rokem obohacují trh. Aby byla alespoň hrubá orientace byly rozděleny na čtyři třídy: domácí počítače, osobní počítače, přenosné počítače, příruční počítače (viz /5/). Představiteli domácích počítačů jsou ZX-81, ZX Spectrum, Commodore 64, (sem bychom mohli zařadit i chystaný čsl. mikropočítač ONDRA). Mezi osobními mikropočítači doslova kraluje IBM PC. Z čsl. trhu bychom sem mohli zařadit PP-06. Mezi osobní mikropočítače patří i naše mikropočítače IQ 151 a PMD-85, které však mají k vlastnostem IBM PC dosti daleko. V roce 1984 v kategorii přenosných počítačů byl jako nejlepší ohodnocen mikropočítač Compag plus. Charakteristickým jevem je vestavěný display a přehrávač pružných disků, hmotnost 13 kg. Představitelem příručních mikropočítačů jsou Sharp PC-5000, či Epson HX-20. Tuto skupinu charakterizuje velikost A5 až A4, schopnost pracovat nezávisle

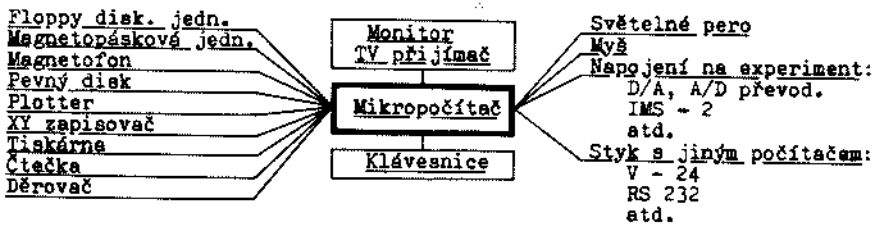
na elektrické síti, tedy z baterií, a to až stovky hodin. Bateriový provoz je samozřejmě i u periferií. Obsah paměti počítače se vypnutím nezničí. Jako zobrazovací jednotka slouží zobrazovač z kapalných krystalů. Provoz na baterie a „nepřetržitá paměť" charakterizuje i další podskupinu - kapesní mikropočítače (např. SHARP PC-1211, PC-1500 aj.)

Počítač, či mikropočítač však není pouze více či méně líbivá skříňka plná integrovaných obvodů. Toto technické řešení (hardware) se neobejde bez programového vybavení (software). Z prvotního extrému, kdy cenu určovalo pouze technické vybavení, rostl podíl ceny programového vybavení (u řídících počítačů až 70% celkové ceny systému). Historicky první programy byly v binárním kódu, později též assemblery a od poloviny padesátých let tzv. vyšší programovací jazyky. Jmenujme FORTRAN (FORmulace TRANslator - překladač vzorečků, formulí), ALGOL (ALGOrytmic Language - jazyk algoritmů), COBOL (COmmon Business Oriented Language - společný jazyk obchodně orientovaný), PL-1, PASCAL, BASIC, SIMULA, FORTH, LOGO (graficky orientovaný jazyk vhodný pro děti) aj.

Mikropočítače se v počátcích z jazyků velkých počítačů mnoho nepoučily. V roce 1973 vyvinuli J. Kemeny a T. Kurz programovací jazyk BASIC (Beginner´s All - purpose Symbolic Instruction Code), který byl po několik dalších let převládajícím programovacím jazykem mikropočítačů. Trvalo poměrně dlouho, než byly napsány první provozní systémy a bylo užito universálnějších jazyků. V roce 1979 Gary A. Kildall předložil operační systém CP/M /103/, který se stal světovým standardem pro 8 i 16-bitové mikropočítače. Paradoxní je, že firma Intel, u které byl Kildall zaměstnán, tento systém odmítla. Uživatelské programy na základě CP/M byly universální, přenositelné na jiné mikropočítače. Počátkem roku 1980 vznik-

lo mnoho firem, zabývajících se pouze programovým vybavením mikropočítačů: Microsoft, Digital Research, Visi Corp aj. V posledních letech se prodíralo na svět strukturované programování a bez BASICovské nostalgie si již vydobilo své místo - např. PASCAL, Micro Prolog, LOGO aj.

Mikropočítače-to není jenom technické a programové vybavení, to jsou i doplňková zařízení - periferie (tiskárny, plottery, disky atd.). I tyto periferie prožily bouřlivý technický i cenový vývoj. V této úvodní kapitole však nechám periferie mikropočítačů stranou.



Obr. 1.1. : Některé periferie mikropočítače

Seriózněji informace ohledně počítačů ve škole předkládají psychologové. /35/. Upozorňují, že může dojít k přesycenosti, monotónnosti, stereotypu při užívání počítače, zmnožuje se počet aktivit žáka, může dojít k přenosu některých omezujících faktorů počítače na žáka, komunikace s počítačem se provádí pomocí jiných jazyků, jiných signálů, atd.

Nyní se opět vrátíme k samotným mikropočítačům.

Většinou se tento současný přehled týkal mikropočítačů západní produkce. Jak vypadá situace v ČSSR? Ke konci roku 1983 se objevily první československé mikropočítače IQ 150, PMD-85 aj. a

též několik mikropočítačových stavebnic např. TEMS 80-03, PMI-80 aj. Jmenujme pro srovnání další čsl. mikropočítače: SAPI 1, PP-01, PP-02 až 04, TNS-64, Didaktic alfa, SAPI-80, CPU 800, ze stavebnic ještě např. MIKROSAT, TEMS 8000 PAS, ŠMS aj. Vidíme, že i u nás se objevilo velké množství mikropočítačů. Tyto čsl. mikropočítače byly určeny jak pro průmyslové aplikace, tak pro vývoj, tak pro cvičné a školní užití. Charakteristické je, že tyto mikropočítače, tak jako v začátcích i ve světě, nejsou téměř s ničím kompatibilní (!). Některé mikropočítače se vyznačovaly větší poruchovostí a objevil se názor, že i toto bylo pozitivní, neboť vyčlenilo opravdové příznivce od zvědavců.

Na závěr této kapitoly ještě SCIFI odstavec. Superpočítačem se může nazvat ten počítač, který zvládne sto milionů operací s plovoucí desetinou čárkou za sekundu. Firma Fujitsu v r. 1984 uvedla model FACOM VP-50, který dokázal 250 milionů výpočtů za sekundu. Americká společnost Cray předvedla model M-XP, který umí 630 milionů výpočtů za sekundu. Roční výroba 15 ks převážně pro armádu, cena 10-12 milionů dolarů. Světový rekord dosud drží firma Fujitsu s modelem FACOM VP-400, který zvládne 1,14 miliard operací za sekundu. Pro srovnání - ani všechna dítka školou povinná, která na naší planetě chodí do školy, by společnými silami neudržela krok s tímto monstrem déle než dvě sekundy (převzato z /11/).

Tato úvodní kapitola měla u čtenáře vyvolat příjemné zadumání nad možnostmi výpočetní techniky či odložit závoj obtížnosti a složitosti.

1.2. Výroba integrovaných obvodů

Mikropočítače jsou plné integrovaných obvodů. To již víme skoro všichni. Avšak jak se takový integrovaný obvod vyrábí, jaké

principy stojí v pozadí jejich výroby, jaké známe druhy integrovaných obvodů ? O této problematice se uživatelé mikropočítačů ve škole a ani v populární literatuře mnoho nedoví. A přesto je právě zvládnutí výroby integrovaných obvodů velmi velké integrace alfou a omegou rozvoje mikropočítačové techniky.

Integrované obvody může rozdělit na analogové (např. zesilovače, stabilizátory aj.) a číslicové (zpracovávají informace ve dvojkovém tvaru). Další povídání bude zaměřeno právě na tuto druhou skupinu. Následuje množství blíže nevysvětlených pojmů, které slouží spíše k informační orientaci. Více se lze dozvědět např. v /33/ aj.

Podle množství elektronických prvků realizovaných na základní destičce (čipu; angl. chip) rozlišujeme několik skupin:

1. Obvody SSI (Small Scale Integration), obvody malé integrace. Na jednom čipu je realizováno až 10Ø elektronických součástek (př. MH 74ØØ, MH 74Ø4, MH 7474 aj.)

2. Obvody MSI (Medium Scale Integration), obvody střední integrace, obsahující na jednom čipu 1ØØ až 1ØØØ součástek (př. MH 749Ø, aritmeticko-logické jednotky aj.)

3. Obvody LSI (Large Scale Integration), obvody velké integrace, obsahující až 1Ø.ØØØ součástek na jednom čipu o ploše asi 1ØØ mm^2 (př. mikroprocesor, paměti aj.)

4. Obvody VLSI (Very Large Scale Integration), obvody velmi velké integrace. Na jednom čipu je realizována např. paměť 1 mega bit, či 16-bitový mikroprocesor.

Další možné rozdělení číslicových integrovaných obvodů je na bipolární a unipolární. U bipolární technologie (základem je bipolární tranzistor) vznikla řada technik používaných u obvodů SSI a MSI i u LSI - jsou to:

- RTL - odporově vázaná logika
- DTL - diodově vázaná logika
- ECL - emitorově vázaná logika
- TTL - tranzistor-tranzistorová logika (nejčastější)
- TTL-S - Schottkyho TTL logika (rychlejší verse TTL)

U bipolárních obvodů se dosahuje nižší hustoty integrace, má poměrně velký ztrátový výkon, je poměrně rychlé.

Integrované obvody unipolární (jejich základem je unipolární tranzistor (MOS)). Z hlediska technologie výroby je tento stavební prvek jednodušší. Unipolární obvody umožňují daleko vyšší integraci, mají menší spotřebu, nevýhodou je nižší rychlost. Technologie výroby unipolárních obvodů:

- PMOS - využívá P kanálu, zavedena v r.1964, neperspektivní, malá rychlost
- NMOS - využívá N kanálu, 3x větší rychlost než PMOS
- CMOS - (Complementary MOS) využívá tranzistory s kanálem P i N, nízký ztrátový výkon
- VMOS - (Vertikal MOS), užívá třírozměrného uspořádání
- HMOS - zdokonalení NMOS, užívá se např. elektronová litografie
- SOS - základní destičku tvoří safírová podložka, velmi drahá technologie, užívá se v extrémních podmínkách (kosmonautika aj.)

Od výčtu druhů a technologií výroby integrovaných obvodů se pokusíme nahlédnout přímo do výroby, jak se vlastně vyrábějí. Základním materiálem je křemík. Křemičitý písek se nejprve čistí a poté roztaví. Tažením z taveniny se získávají monokrystaly křemíku. Monokrystaly mají průměr 5Ø až 8Ø mm a dotují se příměsemi, které způsobí vodivost typu P či N. Poté se rozřežou na tenké destičky (2ØØ až 35Ø μm) a brousí se a leští do rovného povrchu. Z dalších technologických operací se používá epitaxe, difúze, leptání, oxidace. Tyto operace probíhají při velkých teplotách

(12ØØ°C-14ØØ°C) v uzavřeném prostoru syceném parami dotujících prvků. Pro vytvoření vrstvy SiO_2 je užívána oxidace. Při teplotě 9ØØ°C až 12ØØ°C je do prostoru s křemíkovou destičkou přiváděn kyslík či vodní pára. Okysličování probíhá na úkor křemíku. Vrstva SiO_2 slouží jako maska pro další difúzní kroky, jako izolační materiál či dielektrikum pro kondenzátory.

Před každou výše uvedenou metodou se však na destičku musí přenést vlastní geometrická struktura obvodu, která tvoří funkci obvodu. Pro přenesení této struktury se používá speciálních litografických technik jako je fotolitografie či v poslední době elektronová litografie (kdy strukturu obvodu „ryje" do destičky elektronový svazek, který může být velice tenký a umožní zvětšit hustotu součástek na čipu). Fotolitografie je typickou technologií při výrobě integrovaných obvodů.

Vlastní elektronické zapojení-ať již jde o aktivní prvky či spoje-se na základní destičce vytváří několika fotomaskami. Fotografické matrice se zhotovují dle elektrického zapojení a dle technologického návrhu obvodu. Prvotní matrice, které jsou určeny např. pro zhotovení čipu velikosti 2x2 mm, mají velikost nástěnné mapy (např. 2x2 m). Takovýchto matric je celá série. Dále se tyto matrice fotograficky zmenšují.

Vlastní přenesení jedné masky na křemíkovou základní destičku se provede fotochemickou cestou. Destička se polije tenkou vrstvou fotorezistu. To je látka, která při osvětlení ultrafialovým světlem vytvrdne (dojde k polymeraci fotorezistu). Kdo zná výrobu tištěných spojů fotografickou cestou, pozná hned analogii. Avšak fotorezist na výrobu integrovaných obvodů musí mít mnohonásobně vyšší rozlišovací schopnost. (Proto se u složitých obvodů používá elektronové litografie - pro srovnání elektronový litograf by celou naši republiku, kde by byl nejmenší detail ulice

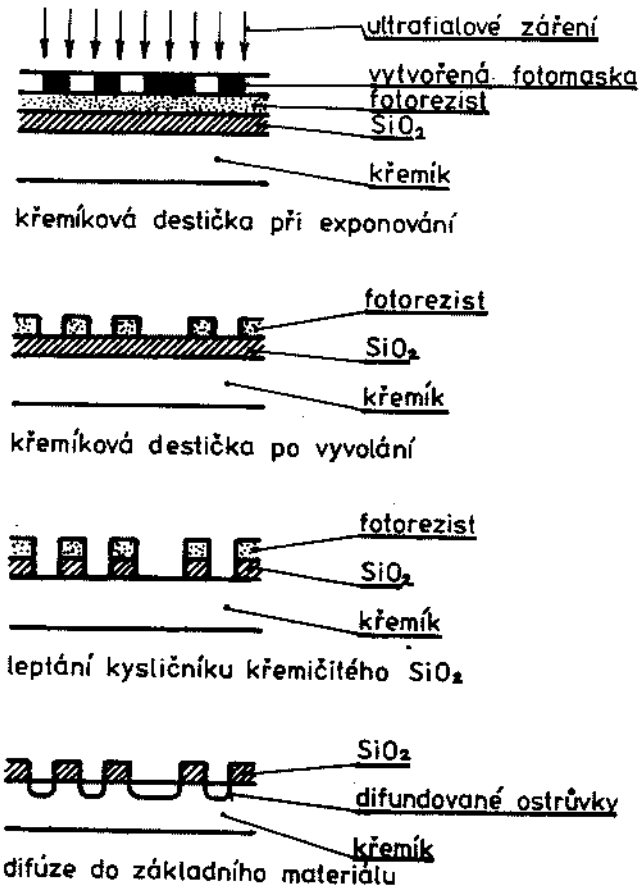
v městě, zobrazil na destičce 10x10 cm; ČSSR patří mezi výrobce těchto špičkových zařízení). Poté se na destičku s fotorezistem přiloží velice přesně fotografická matrice. Následuje expozice ultrafialovým světlem. Tam, kde světlo proniklo matricí, tam dojde k polymeraci fotorezistu. Dále se destička s exponovaným fotorezistem vyvolá. Polymerovaný fotorezist zůstane na destičce.

Nyní se již může provádět chemické „obrábění" těch míst, kde fotorezist není. Nejdříve se odleptá (např. kyselinou fluorovodíkovou) vrstvička SiO_2. Po leptání se odstraní zpolymerovaný fotorezist, který až dosud chránil požadovaná místa dle masky. A nyní se může přistoupit k difúzi příměsových atomů přes otvory vzniklé leptáním v SiO_2 masce. Popsané operace jsou zjednodušeně zobrazeny na obr.1.2.

Propojení jednotlivých funkčních celků či aktivních prvků se provede opět maskou a poté napařením např. hliníku. Po odleptání zůstanou požadované spoje.

Výše provedené operace se neprovádí pro každý čip zvlášť, ale na tenkou křemíkovou destičku (o průměru 5Ø až 8Ø mm) se současně provádějí všechny tyto operace na stovkách čipů. Mezi těmito mnoha stejnými čipy je několik testovacích obvodů, které se po jednotlivých operacích vždy přezkušují. Jetliže se v průběhu výroby zjistí na těchto testovacích obvodech chyba, další operace se neprovádějí a celá destička se vyřadí. Výtěžnost při výrobě integrovaných (tj. množství kompletně funkčních integrovaných obvodů z celkového množství) je tajné číslo všech výrobců integrovaných obvodů, neboť u složitých obvodů je relativně malá. (Jistá firma v Singapuru uvedla, že výtěžnost u mikroprocesorů se pohybuje mezi 1Ø až 15%).

Proběhnou-li všechny operace s křemíkovou destičkou dobře,



Obr. 1.2. : Některé operace při výrobě integrovaných obvodů

pak se tato destička rozřeže na jednotlivé čipy velikosti např. 3x3 mm. Tento čip se poté zapouzdří nejčastěji do známých pouzder z epoxydových hmot či keramiky. Ještě před zapouzdřením se musí tenkým zlatým drátkem propojit kontaktní plošky na čipu s vývody integrovaného obvodu.

Celá výroba integrovaných obvodů probíhá v pečlivě čistých výrobních halách. Lidé pracují v pláštích, s rouškami a úplně nejlépe, když za ně pracují automaty. Pro představu při výrobě paměti 64KB je potřeba docílit čistoty 1 částečka o velikosti Ø,1 mikrometru v 1 dm^3. Pro superpaměti 1Mbit pak superčisté prostředí dosahující čistoty 1 částečky o velikosti Ø,Ø5 mikrometru na 3Ø dm^3. Plocha takovéhoto superčipu je 7x7 mm a odstupy mezi jednotlivými prvky dosahují několik mikrometrů.

Po přečtení této kapitoly by v čtenáři mělo zůstat jakési „pozadí" z výroby integrovaných obvodů a i to, že přechod k „hustší" integraci zaměstnává celou řadu fyziků, inženýrů, technologů. Čtenář by si měl vytvořit představu, že vlastní čip je realizován na velmi malé plošce. Zapouzdřený čip (integrovaný obvod) je mechanicky dosti odolný, avšak „zranitelnost" mikroelektronických součástek (např. EPROM paměti) a tím i celých počítačů může způsobit např. gama záření, či rychlá kosmická částice, která si to „zamíří" na čip paměti a přemaže její obsah. Rozdělení integrovaných obvodů a technologií si neklade za cíl úplnost. Jsou zde spíše uvedeny názvy a zkratky, se kterými se čtenář asi nejčastěji setká.

1.3. Základní pojmy

Čtenář nechť neočekává od této kapitoly podobu encyklopedického slovníku mikropočítačové problematiky. Vybrané pojmy jsou

jakousi úvodní „rozcvičkou" a jsou spíše připomenuty než podrobně objasněny.

Mikropočítač můžeme definovat jako počítačový systém, jehož základem je mikroprocesor.

Mikroprocesor - integrovaný obvod tvořící základní jednotku počítače.

Bit - nejmenší jednotka informace (místo obsahující buď Ø nebo 1)

Byte - (čti bajt) - též slabika - skládá se z osmi bitů

Mikropočítač zpracovává dva druhy informací - data a instrukce. U 8-bitových mikropočítačů je šíře dat 8bitů. Data mohou být např. číselné hodnoty, instrukce sice má podobu číselnou, avšak mikroprocesor ji chápe jako příkaz, co má vykonat (též bude vysvětleno později). Mikropočítače uvnitř pracují v binárním kódu (s nulami a jedničkami). Vyšší úroveň komunikace s mikropočítačem je v hexadecimálním kódu (tu mají např. často mikropočítačové stavebnice) např. při programování ve strojovém kódu. Decimální kód známe všichni a používají ho i vyšší programovací jazyky. Existují i jiné kódy - např. pro záporná čísla aj.

Příklad:	decimální kód	binární kód	hexadecimální kód
	Ø	ØØØØ ØØØØ	Ø
	1Ø	ØØØØ 1Ø1Ø	A
	255	1111 1111	FF

Počítače tedy pracují v binární soustavě. Že dokáží zpracovávat číselné údaje, to nás jistě nezarazí. Ale jak to, že umí zpracovávat také písmena či jiné znaky, které s binární soustavou nemají nic společného? To se provede kódováním; nejznámější je ASCII (zkratka American Standard Code for Information Interchange) kód (7bitový) nebo z dálnopisné techniky CCITT 2 (5bitový). ASCII tabulka je uvedená v příloze. Jenom několik ukázek na ožití:

Příklad:	znak	ASCII kód (decimálně)	ASCII kód (hexadecimálně)
	Ø	48	3Ø
	1	49	31
	⋮	⋮	⋮
	A	65	41
	B	66	42

Jestliže má ASCII kód 7bitů pak můžeme zakódovat 128 znaků. V IQ 151 je využito všech 8bitů a tak je umožněna i inverse znaků. (Při inversi je nejvyšší bit, tj. bit číslo 7, roven "1", jinak je roven "Ø". Pozn. - dále v textu budeme hexadecimální číslo označovat písmenem H za příslušným číslem (např. 3ØH je hex. číslo 3Ø).

Adresa paměťového místa - každé paměťové místo má přesně určenou polohu, kam může mikroprocesor uložit, či naopak kde může vyzvednout informaci. Rozsah adresace u 16-bitové adresy je 64KB (či FFFF hexadecimálně). Mikropočítač dovede komunikovat též se vstupními a výstupními porty, což jsou obvody rozhraní mezi mikroprocesorovým systémem a dalšími zařízeními. Z hlediska programátora fungují vstupní a výstupní porty obdobně jako paměťová místa. Adresa portu je jednobytová (ØØH až FFH)

Dekodér - obvod, který slouží k převodu kódovaných signálů z jednoho kódu do jiného kódu.

Registr je velmi často užívaný pojem. Je to paměť osmi bitů (např. přímo v mikroprocesoru), kam může uživatel či pouze mikroprocesor uložit či vyzvednout byteovou informaci. Registr může být posuvný, tj. informaci lze posunout doleva či doprava, registr můžeme naplnit či přečíst najednou (paralelní vstup či výstup registru) nebo postupně bit po bitu (sériový vstup či výstup registru).

Klopný obvod - sekvenční obvod se dvěma stabilními stavy, je základem paměťových obvodů.

Hodiny (Clock) - jakési hnací péro v mikropočítači. Bez nich by se neuskutečnila žádná činnost. Hodinový signál synchronizuje jednak mikroprocesor a jednak spolupráci mikroprocesoru s ostatními částmi mikropočítače. Kmitočet hodin nemůže být nulový, neboť mikroprocesor je dynamická struktura (obdoba dynamické paměti RWM - vysvětl. později; při nulovém kmitočtu by ztratila informace).

Uložení číselné proměnné (do 4 byte) a uložení stringové proměnné je pro IQ 151 popsáno v /14/.

Rychlost mikropočítače není dána jenom kmitočtem hodin, je dána „chytrostí" mikroprocesoru a ta je zase dána instrukčním souborem.

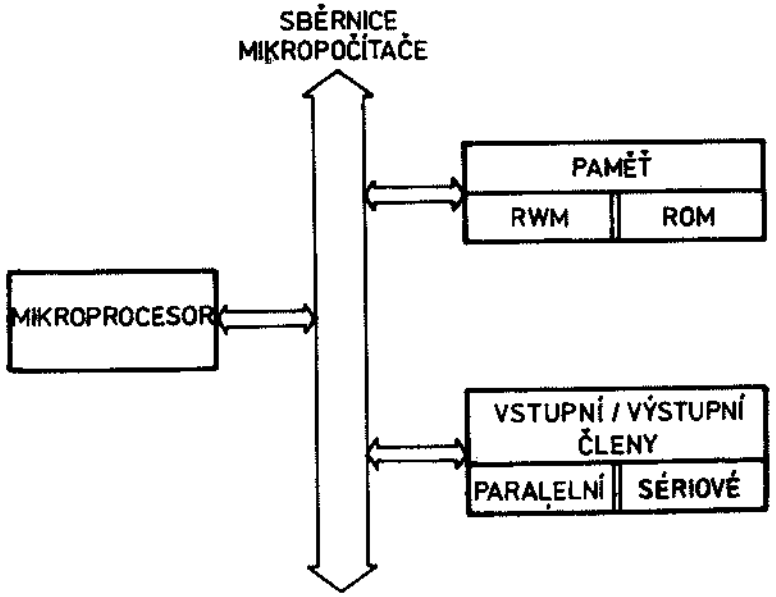
Některé další pojmy budou postupně přibírány a vysvětleny, další by měly být objasněny např. v /14/, /15/, /16/, /17/, /12/ aj. Zvláště bychom chtěli doporučit /36/, což je středoškolská učebnice zaměřené na číslicovou techniku-Boolovou algebrou počínaje, přes klopné obvody, až po paměti počítače.

2. Struktura mikropočítače

Základní struktura mikropočítače je obdobná jako struktura ostatních počítačů. My se zaměříme na počítače tzv. „von Neumannova typu", které mají společnou paměť jak pro data, tak pro programy (na rozdíl od tzv. harvadské koncepce počítače). Mikropočítač zpracovává data postupně po bytech, tedy tuto činnost můžeme označit jako „sériovou". Popíšeme universální strukturu mikropočítače s mikroprocesorem.

2.1. Skupinové schema mikropočítače

Celková obecná sestava mikropočítačového systému (např. s mikroprocesorem 8080) je zjednodušeně znázorněna na obr.2.1.



Obr. 2.1.: Skupinové schema mikropočítače

Blok označený mikroprocesor obsahuje kromě vlastního mikroprocesoru (např. typu 8080) též některé jeho tzv. podpůrné obvody (např. generátor hodinových impulsů - typ 8224 a tzv. systémový řadič - typ 8228), případně též přizpůsobovací členy pro připojení na paměť, resp. na vstupní a výstupní členy. Některé modernější mikroprocesory (např. typ 8085 či Z-80 aj.) mají již množství podpůrných obvodů minimalizováno. Mikroprocesor řídí celou činnost mikropočítače. Zajišťuje správné provádění instrukcí, zpracování a tok dat aj. Autor /12/ ho nazývá dirigentem orchestru.

Paměť mikropočítače obsahuje paměť s pevným obsahem - ROM (Read Only Memory) tak i paměť pro zápis a čtení - RWM (Read/Write Memory) programů a dat (někdy též označovanou nepřesně RAM).

Vstupní / výstupní členy tvoří obvody, které umožňují napojení mikropočítače na periferní zařízení (klávesnice, tiskárna aj).

Sběrnice mikropočítače tvoří jakousi páteř mikropočítačového systému. Je rozvedena do všech důležitých bloků mikropočítače. Tvoří ji jednak adresové sběrnice (např. u mikroprocesorového systému 8080 má 16 vodičů), dále datová sběrnice - u 8-bitového počítače má 8 vodičů a dále řídící sběrnice, která má 5 vodičů a zajišťuje řízení činnosti pamětí, vstupně/výstupních členů a přerušovacího systému (objasníme později).

Z obr.2.1. se nám jistě jeví struktura mikropočítače až příliš jednoduchá. Pilný čtenář se může podívat na strukturu konkrétního mikropočítače IQ 151 na obr.4.1., který je detailnější oproti obr.2.1.

Nyní probereme podrobněji základní bloky mikropočítače.

2.1.1. Mikroprocesor

Mikroprocesor je základní jednotka mikropočítače realizovaná na jednom čipu jako LSI obvod. Obsahuje zpravidla řídící jednotku, aritmeticko-logickou jednotku, několik universálních registrů a obvody vstupu a výstupu sběrnice.

Mikroprocesory prošly bouřlivým rozvojem nejenom v šíři dat od 4-bitových až po 32-bitové, ale hlavně ve vývoji účinného souboru instrukcí. Uveďme si několik typů mikroprocesorů, které doznaly velkého rozšíření a možno říci „světového standardu".

Intel-4004: (rok výroby 1971), technologie PMOS, nízký hodinový kmitočet, jednoduchý instrukční soubor, délka slova 4-bity

Intel-8008 (rok výroby 1972), technologie PMOS, adresovatelný prostor 16KB, délka slova 8-bitů, instrukční soubor-48 instrukcí, 7 pracovních registrů, 1 přerušovací vstup, instrukční cyklus 20 μs, RVHP výrobce - NDR

Intel-8080 (rok výroby 1973), technologie NMOS, adresovatelný prostor 64KB, délka slova 8-bitů, instrukční soubor - 78 instrukcí, 7 pracovních registrů, 1 přerušovací vstup, rychlost vykonání instrukce 2÷10 μs, RVHP výrobce - ČSSR, SSSR, MLR

Zilog-Z-80 (rok výroby 1976), technologie NMOS, přímo adresovatelný prostor 64KB, délka slova 8-bitů, instrukční soubor - 158 instrukcí, 7+7 pracovních registrů, 1 přerušovací vstup, základní instrukční cyklus 1 μs, výstup refreš pro dynamické paměti, RVHP výrobce - NDR. Pozn. refreš slouží k obnovování obsahu dynamických pamětí.

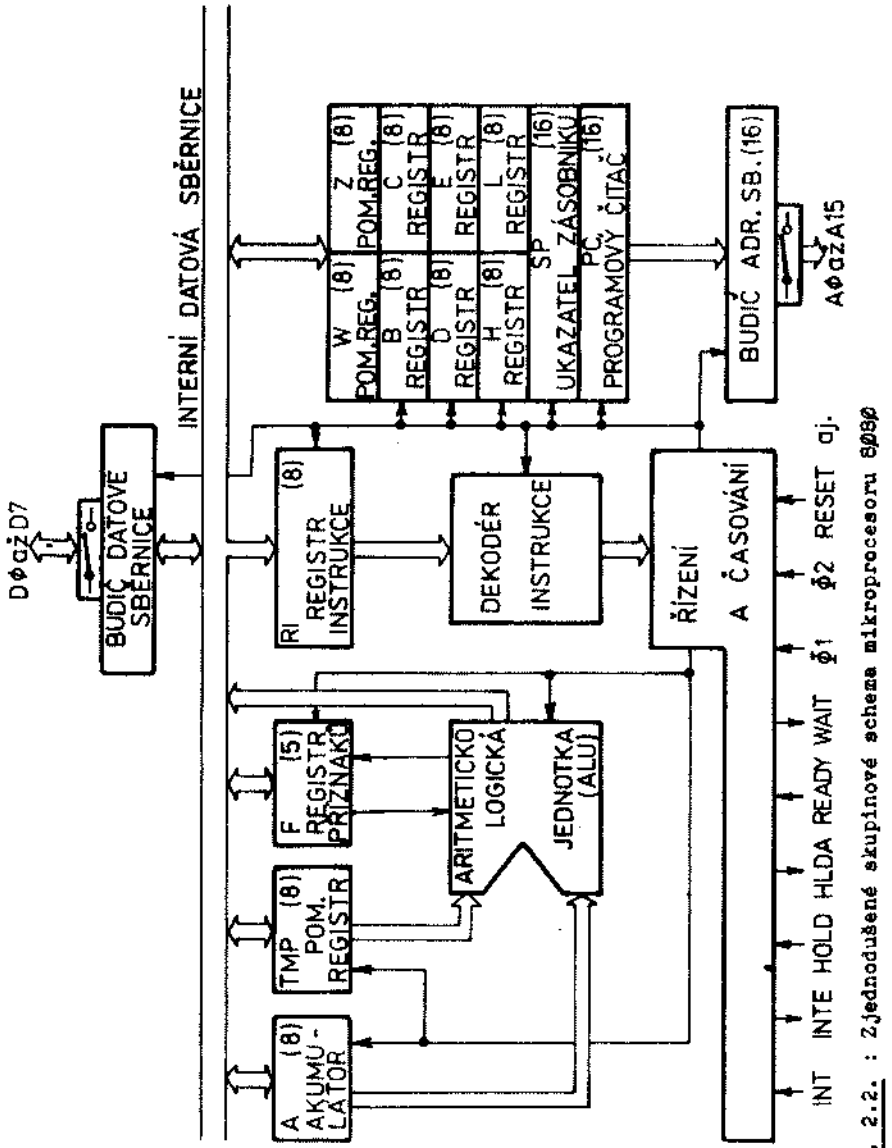
Intel-8Ø86 (rok výroby 1978), technologie HMOS, adresovatelný prostor 1MB, délka slova 16-bitů, instrukční soubor 97 instrukcí, 7+4 pracovních registrů, 1 přerušovací vstup, základní instrukční cyklus Ø,8 μs, RVHP výrobce - SSSR

Motorola-M 68ØØØ (rok výroby 1978), technologie HMOS, adresovatelný prostor 16MB, délka slova 16-bitů/ vnitřně 32 bitů, 16 pracovních registrů, instrukční soubor 63 instrukcí (ovšem na úrovni minipočítače), 1 přerušovací vstup, základní instrukční cyklus Ø,5 μs

V našem dalším popisu se podrobněji zaměříme na mikroprocesor typu 8Ø8Ø, který je též užit v československých mikropočítačích IQ 151, PMD-85, SAPI-1 aj.

Mikroprocesor se může začátečníkovi jevit jako velice složité a téměř nepochopitelné zařízení. Avšak uživatel této složitosti vůbec nevyužije nebo spíše tuto konstrukční složitost využije sekundárně jako jednoduchost, snadnost obsluhy mikroprocesoru. Mikroprocesor je obdařen malou „umělou inteligencí" či spíše složitým algoritmem, který se nám automaticky stará o správné provedení požadované činnosti o převzetí hodnot, o uložení výsledků, o časování aj. Uživatel pouze předepíše mikroprocesoru požadovanou činnost (tj. zvolí nějakou instrukci z instrukčního souboru mikroprocesoru) a mikroprocesor tuto činnost předepsaným způsobem - tzv. mikroprogramem vykoná. Do mikroprogramu nemá uživatel přístup, ten je hardwarově vytvořen v mikroprocesoru.

Na obr.2.2. je uvedeno zjednodušené skupinové schema mikroprocesoru (typ 8Ø8Ø). I toto zjednodušené schema se začátečníkům jistě jeví jako složité. (Pokročilé odkazujeme např. na /30/). Pro začátečníka bude postačující základní představa mikroprocesoru



Obr. 2.2. : Zjednodušené skupinové schema mikroprocesoru 8Ø8Ø

jakožto registrové struktury. Dále by měl vědět o aritmeticko-logické jednotce a o bloku řízení a časování.

Nyní trochu podrobněji ke struktuře a činnosti mikroprocesoru 8Ø8Ø. Soubor registrů v mikroprocesoru je konstrukčně tvořen statickou pamětí RAM-RWM. Jednotlivé registry:

- programový čítač (PC), (16bitů)
  Je automaticky zvětšen o jednotku (inkrementován) po každém čtení instrukce z paměti, slouží k adresování („vedení") programu.
- ukazatel zásobníku (SP), (16bitů)

  Soubor registrů není příliš bohatý. Proto se v části hlavní paměti RWM vytvoří tzv. zásobníková paměť či zásobník. K adresování zásobníku slouží ukazatel zásobníku (SP). Při ukládání do zásobníku se ukazatel zásobníku zmenšuje, při čtení se naopak zvětšuje. Podrobněji o zásobníkové struktuře též /15/.
- universální 8-bitové registry B,C,D,E,H,L (lze sestavit i do párů). K souboru registrů lze též formálně přiřadit i tzv. fiktivní registr M umístěný mimo mikroprocesor v hlavní paměti RWM mikropočítače.
- pomocné registry W a Z, (8 bitů), nejsou programově dostupné, slouží např. pro zápis druhého a třetího byte instrukce (př. instrukce C3 ØØ F8)
- akumulátor (A), (8bitů), často užívaný registr, spolupracuje s aritmeticko-logickou jednotkou, uložení výsledků mnohých operací
- pomocný registr (TMP), (8bitů), spolupracuje s aritmeticko-logickou jednotkou
- registr příznaků (F), (5bitů), spolupracuje s aritmeticko-logickou jednotkou, obsahuje známé příznakové bity S - znaménko výsledku operace, Z - nulový výsledek operace, CY - přenos z nejvyššího řádu, AC - přenos mezi bity č.3 a 4, P - parita výsledku (podrobněji již ve /15/)
- registr instrukce (8bitů) přijme operační znak každé instrukce (její první slovo) (např. JMP) a je k dispozici dekodéru instrukce a po dekódování posléze i řízení a časování mikroprocesoru.

Aritmeticko-logická jednotka (ALU) provádí aritmetické, logické a posuvné operace, výsledek operací ovlivňuje stav příznakových registrů. ALU obsahuje např. též hradlo AND či OR , které jsou základem při logických instrukcích AND či OR. Pomocí posunů 8-bitové informace vlevo či vpravo např. o jeden bit se v ALU realizuje násobení resp. dělení dvěma. Takovýchto konstrukčních řešení pro aritmetické a logické operace obsahuje ALU ještě více. Stačí navolit požadovanou činnost (zajistí dekodér instrukce) a ALU provede příslušnou činnost na operandech připravených v registrech A a TMP . Výsledek může být předán např. na vnitřní datovou sběrnici.

Budič datové sběrnice je obousměrný, třístavový, izoluje vnitřní datovou sběrnici mikroprocesoru od vnější systémové datové sběrnice DØ až D7.

Budič adresové sběrnice je jednosměrný, třístavový, vytváří systémovou adresovou sběrnici AØ až A15. Pozn. - schematické vypínače u budičů datové a adresové sběrnice mají upozornit, že platné adresy a data jsou na sběrnicích pouze velmi krátkou dobu.

Zmíníme se ještě o některých signálech z řízení a časování mikroprocesoru:

- INT - je vstup signalizující mikroprocesoru žádost o přerušení programu
- INTE - je výstup signalizující, zda je povoleno přerušení
- HOLD - je vstup signalizující mikroprocesoru požadavek odpojení (uvedení do třetího stavu) datové a adresové sběrnice.
- HLDA - je výstup potvrzující akceptování požadavku HOLD
- RESET - je vstup vyvolávající mimo jiné nastavení registru PC=Ø
- $\phi 1, \phi 2$ - dvoufázový hodinový signál

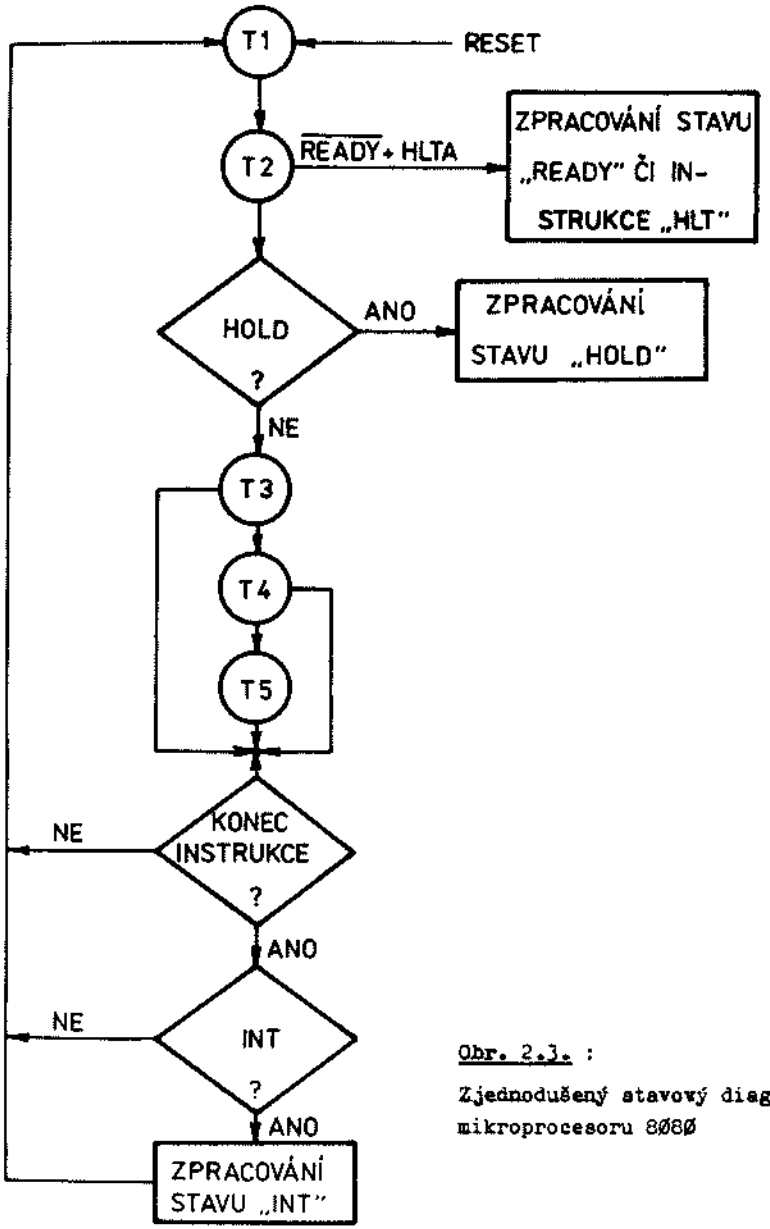
Základní činností mikroprocesoru je vykonání instrukce z instrukčního souboru daného mikroprocesoru. Každé instrukce sestává z jednoho až pěti taktů (M1 až M5) (dle instrukce). Dále každý takt se skládá ze tří až pěti dob T1 až T5, přičemž každá doba trvá právě jednu periodu hodin (obvyklá hodnota periody hodin-asi 500 ns). Např. takt se čtyřmi dobami trvé 2 μs, nejdelší instrukce XTHL má 5 taktů s celkem 18 dobami (9 μs). Počet různých taktů je deset:

1. čtení operačního znaku instrukce
2. čtení z paměti
3. zápis do paměti
4. čtení zásobníku
5. zápis do zásobníku
6. čtení vstupního portu (brány)
7. zápis do výstupního portu (brány)
8. přijetí žádosti o přerušení
9. provedení instrukce HLT
10. přijetí žádosti o přerušení ve stavu HALT

(začátečník si je nemusí pamatovat!)

Typ taktu, který proběhne oznámí mikroprocesor tzv. stavovým slovem vždy v první době (T1) taktu.

Činnost mikroprocesoru v jednotlivých dobách T1 až T5 znázorňuje zjednodušený stavový diagram mikroprocesoru (8080) na obr.2.3. Víme již, že každý takt sestává (dle instrukce) ze tří až pěti dob T1 až T5. Záměrně je zdůrazněna lineární větev stavového diagramu, která je nejběžnější. Na obr.2.3. je vynechána větev týkající se instrukce HLT a stavu při aktivním signálu READY a též stavu HOLD. Ze stavového diagramu je účelné si zapamatovat, že požadavek na HOLD se testuje v době T2 (tedy uvnitř instrukce (!)) a požadavek na přerušení (INT) se testuje po skončení instrukce (!). Jinak každý takt prochází postupně dobami T1 až T3, resp. T1 až T5, a testem na INT a HOLD (platí pro lineární větev).



Obr. 2.3. :
Zjednodušený stavový diagram mikroprocesoru 8080

Činnost mikroprocesoru je definovaně zahájena aktivací signálu RESET (PC se nastaví na Ø), která se provede automaticky po zapnutí mikropočítače či manuálně např. z klávesnice.

Rozebraná ukázka realizace strojové instrukce v mikroprocesoru je v kap.3.1.2. Podrobně je každá instrukce rozebrána do všech taktů a dob např. v /16/.

Shrnutí základních poznatků o mikroprocesoru 8Ø8Ø

Mikroprocesor MHB 8Ø8Ø A (čsl.značení) má 8-bitovou datovou sběrnici, 16-bitovou adresovou sběrnici a 5-bitovou řídící sběrnici. Jeho časování je dvoufázové a zajišťují ho externí obvody. Mikroprocesor má vestavěnou zápisníkovou paměť se 7 universálními registry (A,B,C,D,E,H,L) a je vybaven instrukčním souborem 78 instrukcí. Zásobníková paměť je vytvořena v části hlavní paměti RWM, v mikroprocesoru je k dispozici pouze 16-bitový ukazatel zásobníku (SP). Další 16-bitový registr tzv. programový čítač (PC) slouží k adresování programové části operační paměti.

Mikroprocesor 8Ø8Ø umožňuje pozastavení činnosti (signály READY, WAIT) při komunikaci s pomalejšími periferiemi, dále je vybaven přerušovacím systémem (signály INT, INTE) a dále též umožňuje odpojení (uvedení do třetího stavu - vysv. viz.2.1.2) datové a adresové sběrnice (např. pro přímý přístup do operační paměti bez činnosti mikroprocesoru (DMA - přenos) (signály HOLD, HLDA).

Činnost mikroprocesoru probíhá v taktech. Nejkratší instrukce sestává z jednoho taktu (4 doby), nejdelší z pěti taktů (18 dob). Každý takt má minimálně 3 doby (T1, T2, T3), některé až 5 dob. Řídící signály pro činnost mikroprocesoru se odvozují od tzv. stavového slova, kterých je u mikroprocesoru 8Ø8Ø deset typů.

Ze základního algoritmu činnosti mikroprocesoru 8Ø8Ø je vhodné též vědět, že přerušení může nastat až po skončení právě probíhající instrukce a stav HOLD i uprostřed instrukce po době T2.

Mikroprocesor 8Ø8Ø není sám schopen činnosti, potřebuje ještě některé podpůrné obvody, které potom s mikroprocesorem vytvářejí tzv. skupinu mikroprocesoru.

Některé podrobnější informace k mikroprocesoru 8Ø8Ø jsou též ještě v kap.4.1. Případné zájemce o podrobnější činnost odkazuji na /16/, /30/,/26/ aj.

2.1.2. Sběrnice mikropočítače

Víme již, že všechny bloky mikropočítače jsou paralelně propojeny souborem vodičů - sběrnicí. V mikroprocesoru ani v mikropočítači není možné propojit každý blok s každým zvláštním vodičem. Proto se přistupuje ke sběrnicovému typu propojení, které představuje velkou materiálovou úsporu. Jenom pro představu v mikroprocesoru 8Ø8Ø (který má samozřejmě interní sběrnicovou strukturu) zabírá sběrnice 80% plochy čipu.

Některé vodiče společné sběrnice jsou určeny pro vlastní přenos dat - datová sběrnice, zbylá část vodičů je vyhražena pro řízení toku dat po datové sběrnici. Jedná se především o adresovou sběrnici a o řídící sběrnici. Adresová sběrnice určuje, kterých paměťových buněk nebo vstupně-výstupních portů se bude přenos dat týkat. Řídící sběrnice určuje směr přenosu a časování.

Jelikož tu samou sběrnici využívají všechny propojené bloky, musí být tato sběrnice každou chvíli přidělena jiným blokům (ať se jedná o bloky, které předávají, či které přijímají informace). Platné informace na datové, adresové a řídící sběrnici pro ten daný přenos jsou na sběrnici velice krátkou dobu. Poté je sběr-

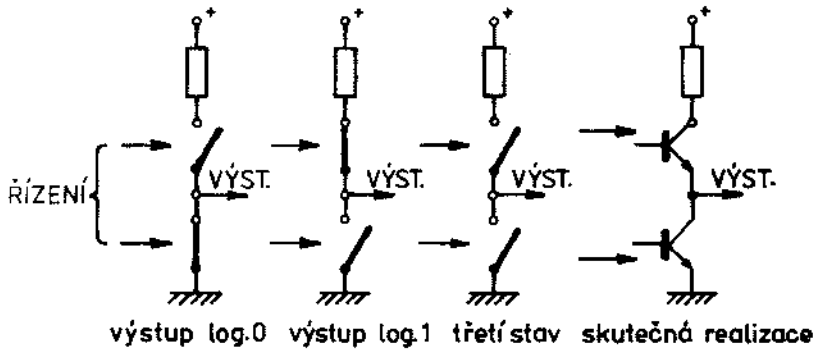
nice k dispozici dalšímu přenosu. Mikroprocesor je výjimečný účastník při všech přenosech (pozn. pro znalejší - výjimku tvoří tzv. DMA-přenos, kdy je „pánem" nad sběrnicí právě DMA obvod).

Obvodová realizace sběrnicového uspořádání, které by umožňovalo řízení v různých časových intervalech a z různých zdrojů, se nejčastěji tvoří z budičů sběrnice s tzv. třístavovým výstupem nebo z budičů sběrnice s otevřeným kolektorem. Princip a realizace třístavového budiče je na obr.2.4. Normální logické výstupy mají stav logická „1", logická „Ø" a tento typ budiče ještě navíc tzv. „třetí stav" (či stav vysoké impedance), či nejnázorněji stav, kdy je budič úplně odpojen od výstupu. Na obr.2.5. je princip a realizace budiče s otevřeným kolektorem. Požadované výstupy budičů mají jeden společný pracovní odpor (externí) a každý výstup může do společného vodiče vysílat libovolný stav log. „Ø" či log. „1" (na rozdíl od přímého logického výstupu, jestliže nejsou užity budiče s otevřeným kolektorem). Na společném pracovním odporu potom vytváření logickou funkci OR („montážní OR"). Na obr.2.6. je ukázka přenosu informace (bitu) po jednom adresovém vodiči. Povšimněte si paralelního připojení několika budičů (třístavových) a několika snímačů. Nechť má v daném okamžiku aktivní vstup řízení budič č.1 a snímač č.2, pak mezi nimi proběhne přenos informace. V daném okamžiku musí být aktivní pouze jeden budič.

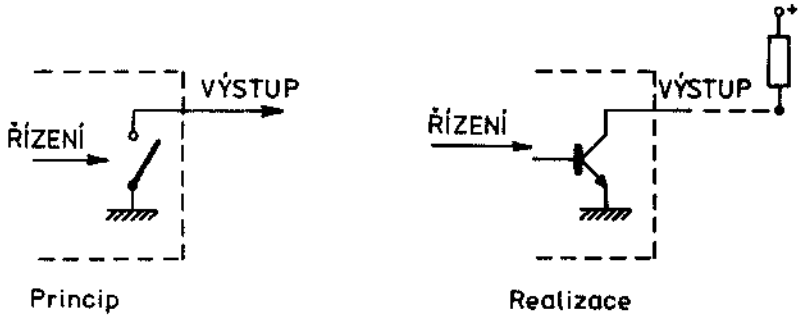
Při normální činnosti mikropočítače se vystačí se čtyřmi základními druhy přenosů po sběrnici:

1) čtení z paměti
2) zápis do paměti
3) čtení vstupního portu
4) zápis do výstupního portu

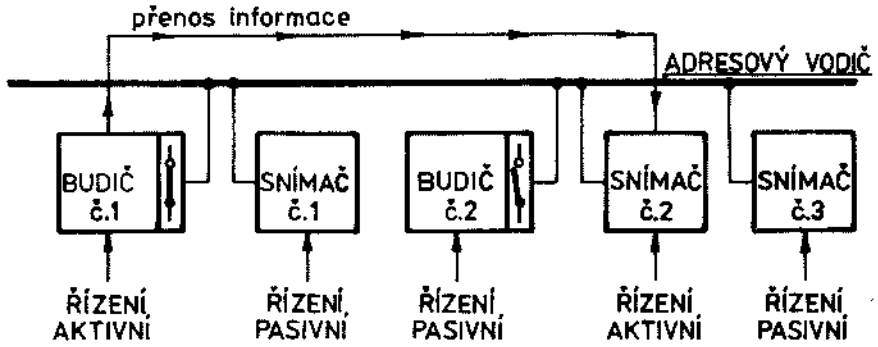
Pro znalejší ještě doplňme dva rozšířené přenosy po společné



Obr. 2.4. : Princip a realizace budiče sběrnice s třístavovým výstupem



Obr. 2.5. : Princip a realizace budiče sběrnice s otevřeným kolektorem



Obr. 2.6. : Ukázka přenosu informace po jednom adresovém vodiči

sběrnici:

5) přerušení

6) přímý přístup do paměti

Řízení těchto přenosů včetně jejich časování ponecháme na čtenáři s doporučením literatury např. /16/ aj. Do řízení přenosů nemůže uživatel zasahovat (může zasáhnout pouze jako tvůrce programu). Celé řízení sběrnice je pod kontrolou mikroprocesoru (pouze výjimka DMA přenos).

Snahou výrobců mikropočítačů i uživatelů je vytvoření standardní sběrnice, která umožňuje podstatně snadnější slučitelnost (ať již obvodovou nebo programovou). Standardních sběrnic existuje více druhů. Jmenujme některé světově rozšířené - např. sběrnice MULTIBUS, S 1ØØ, STD BUS aj.

2.1.3. Paměť mikropočítače

Mikropočítač potřebuje ke své činnosti paměťové buňky (paměti), ve kterých je uložen např. program pro požadovanou činnost, záznam výsledků aj. Samotný mikroprocesor obsahuje pouze několik universálních registrů, (na které můžeme též pohlížet jako na paměťové buňky), avšak činnost nad těmito několika registry by nebyla asi zajímavá. Proto má mikropočítač k dispozici paměť. A čím je tato paměť větší, tím je uživatel spokojenější.

V technice současných mikropočítačů se dnes téměř výhradně používají polovodičové paměti, které předčí jiné druhy pamětí (např. feritové) jak v rychlosti, tak v příkonu, tak v rozměrech, tak v ceně aj.
Jedinou nevýhodou polovodičových pamětí RWM je ztráta informace při odpojeném napájení. Avšak již nyní existují polovodičové RWM paměti, které mají zálohované bateriové napájení přímo vestavěné

ve svém pouzdru atd.

Polovodičové paměti můžeme rozdělit podle několika různých kritérií. Nejčastější kriteria jsou:

1. podle přístupu k datům → SAM, RAM

2. podle modifikace obsahu paměti → RWM, ROM (PROM, EPROM)

3. podle realizace paměťové buňky → statické, dynamické

4. podle základní polovodičové technologie → bipolární paměti, MOS paměti

Z hlediska přístupu k datům rozeznáváme paměti s libovolným přístupem RAM (Random Access Memory), kdy po adresaci máme okamžitě možnost modifikovat tuto adresovanou buňku. Paměti SAM (Serial Access Memory) mají sériový přístup k datům (např. magnetopáskové paměti, diskové paměti aj.). Zde se musí příslušná buňka vyhledávat vždy od počátku souboru.

Pro zajímavost uveďme ještě typ asociativní paměti, která se začíná uplatňovat u superpočítačů. Na rozdíl od paměti RAM není zapotřebí znát umístění dat (adresu), protože u asociativní paměti probíhá vyhledávání dat podle obsahu paměti, na základě určitých aspektů dat.

Z hlediska možnosti změny dat máme paměti RWM (Read Write Memory), které jsou určeny pro čtení i zápis dat, dále paměti ROM (Read Only Memory), které jsou určeny převážně ke čtení dat. Někdy se též tyto paměti ROM vyrábí ve verzi PROM (Programmable

ROM), kdy si sám uživatel může naprogramovat obsah ROM (avšak pouze jednorázově - nevratně), či ve verzi EPROM (Erasable PROM), které si opět může uživatel sám naprogramovat (avšak lze ji působením UV světla (čip této paměti je pod křemíkovým okénkem) vymazat a opět naprogramovat). Paměti EPROM jsou velmi oblíbené pro svoji kapacitu, reprogramovatelnost, stálost naprogramované informace, aj. Pro zajímavost - náboj, který se vytvoří programovacím impulzem se zmenší asi o 30% za 100 let.

Z hlediska realizace základní paměťové buňky rozlišujeme paměti statické, kde je základem bistabilní klopný obvod, a dynamické, kde je jako paměť využita parazitní kapacita tranzistoru MOS. Statické paměti uchovávají informaci po celou dobu, kdy je připojeno napájení. Dynamické paměti vyžadují periodické obnovování zaznamenané informace (Refresh).

K rozdělení podle polovodičové technologie jen poznámka, že bipolární paměti mají vyšší rychlost k přístupu dat, mají větší příkon a menší kapacitu než paměti zhotovené technologií MOS.

U mikropočítačů jsou nejčastěji používané paměti RWM-RAM dynamické, technologie MOS a paměti ROM, či u malých sérií pak EPROM.

Paměť mikropočítače není vytvořena jedním integrovaným obvodem, proto je třeba paměti poskládat do určité organizace (např. Nx8 bitů).

Čtenáře, kteří chtějí znát jak vypadá jedna paměťové buňka, či jak se programují paměti EPROM, či jaká specifika se musí dodržet při návrhu jednotlivých typů paměti, odkazuji na literaturu /33/, /32/, aj. Zde čtenář nalezne informace i o jiných typech polovodičových resp. i nepolovodičových pamětí.

2.1.4. Vstupy a výstupy mikropočítače

To jsou části mikropočítače, které asi uživatele nejvíce zajímají, neboť k čemu by nám byl mikropočítač bez displeje, případně klávesnice aj. Mezi vstupně výstupní periferie počítáme též všechny druhy paměťových médií (tedy např. i magnetofon, případně děrovač, či snímač děrné pásky, floppy disk aj.). Další trochu odlišnou skupinu vstupně výstupních obvodů mikropočítače, umožňující začlenění mikropočítače do reálného řízeného procesu, tvoří převodníky analogového signálu na číslicový, případně převodníky číslicového signálu na analogový. Můžeme k nim ještě přidat různé tvarovače, zesilovače či zařízení sloužící k rozmanité transformaci fyzikálních veličin.

Všechna tato vstupně výstupní zařízení jsou přes své stykové rozhraní (Interface) připojena na společnou systémovou sběrnici mikropočítače. Mikropočítač a na druhé straně periferie tvoří oddělené systémy, které jsou v některých případech schopny i částečně autonomní činnosti. Součinnost mikropočítače a periferií je třeba vhodným způsobem synchronizovat. Pokud požadovaná rychlost zpracování mikropočítačem je vyhovující, pak je mikropočítač pro tyto případy vybaven systémem přerušení, či pro ještě rychlejší blokové přenosy dat pak přímým přístupem do paměti (DMA). Pro běžné aplikace, které nevyžadují rychlé odezvy mikropočítače, je možné vystačit programovým řízením součinnosti mezi mikropočítačem a periferií.

Protože je rychlost počítače a periferie obecně různá, je vzájemná komunikace založena na tzv. korespondenčním styku. Mikropočítač (v případě výstupu dat na periferii) potvrzuje platnost dat přiložených na sběrnici a výstupní periferie potvrdí zpracování dat. Do té doby než jsou data zpracována nemůže mikropočítač přiložit

na sběrnici další data. Vidíme, že tímto jednoduchým způsobem, jehož realizace může být jak technické řešení, případně též programová obsluha, je jednoznačně zajištěn správný přenos dat. V případě, že je mikropočítač jako příjemce vstupních dat, pak platnost dat na sběrnici potvrzuje vysílací periferie a mikropočítač potvrzuje zpracování dat.

Z hlediska elektronických obvodů a postupu předávání dat můžeme rozlišit paralelní a sériový přenos dat či další rozdělení na synchronní a asynchronní přenos dat. Paralelní přenos dat je z konstrukčního hlediska nejjednodušší. Propojení musí být provedeno osmi vodiči datové sběrnice a několika vodiči pro řízení přenosu. Nejčastější realizace je např. neprogramovatelným obvodem MH 3212, či programovatelným obvodem MHB 8255. Příkladem paralelního asynchronního přenosu může být přenos dat na tiskárnu s paralelním rozhraním. Též však existují tiskárny se sériovým asynchronním rozhraním, které snadno odlišíme, neboť jsou s mikropočítačem propojeny dvěma (resp. čtyřmi) vodiči. Uvědomme si, že po těchto dvou vodičích musí jít jak data, tak i řídící povely. Každý byte, který je na této sběrnici, je doplněn tzv. obalovými bity - start bitem, který dává informaci, že budou následovat datové bity Ø-7, a stop bitem (případně několika stop bity), které zase dávají informaci mikropočítači, že přenos jednoho byte skončil. Takováto data - byty - spolu s obalovými bity se na sběrnici objevují v libovolných okamžicích - asynchronně (nejsou svázány s žádnými hodinami systému). Pro sériový styk je výhodné užít obvod MHB 8251, který patří do rodiny programovatelných obvodů a který automaticky převede paralelní byte do sériové podoby včetně obalových bitů a naopak přijatý sériový byte nejdříve ocleští o obalové bity a sériově přijatý byte převede do paralelní formy.

Vážnější zájemce o styk mikropočítače s periferiemi odkazuji na literaturu /19/, /16/, /12/ aj., kde je též podrobně objasněno časování signálů sběrnice, případně jsou popsány některé konkrétní typy obvodů usnadňující realizaci vstupu a výstupu mikropočítače.

2.1.5. Přerušovací systém

Již při výkladu činnosti jsme narazili na pojem přerušení. Systém přerušení umožňuje přerušit (určitým vnějším podnětem) probíhající program a vyvolat důležitější program s vyšší prioritou. Po skončení takového programu s vyšší prioritou se počítač samočinně vrátí k přerušenému programu. Přerušení (jak již víme ze základního algoritmu mikroprocesoru 8Ø8Ø -viz. obr.2.3.) může být zpracováno až po skončení celé instrukce. Vnější podnět, který má vyvolat přerušení, se může připojit přímo na vstup mikroprocesoru INT. Připomeňme si, že mezi základními instrukcemi mikroprocesoru 8Ø8Ø /15/ jsou instrukce pro povolení (EI) či zakázání (DI) přerušení. Obvod MH 8228, který je součástí skupiny mikroprocesoru nám usnadňuje zpracování přerušení. Je-li jeho výstup $INTA^{*}$ připojen přes rezistor 1K Ω na +12V, pak tento obvod generuje při přijetí žádosti o přerušení instrukci RST 7, která jak již víme znamená skok na pevnou adresu 38H, kde bude začínat náš podprogram pro obsluhu přerušení. Pro zpracování více různých priorit je možné využít některé neprogramovatelné obvody např. MH 3214, který má 8 priorit (pro více priorit je možnost je kaskádně řadit), a jeho obslužné podprogramy pro přerušení začínají na pevných adresách ØH, 8H, 1ØH, 18H, 2ØH, 28H, 3ØH, 38H (což odpovídá instrukcím RST Ø až RST 7). Daleko větší možnosti poskytuje přerušovací systém s programovatelným obvodem typu 8259, na

jehož obslužné podprogramy přerušení se přechází instrukcí CALL adresa, kde adresa je zvolí programovacími slovy tohoto obvodu (není tedy pevně určená). Jeden obvod má opět 8 priorit (též je možnost kaskádního řazení). O programování tohoto obvodu a o ostatních možnostech odkazuji vážné zájemce na literaturu /26/. V této literatuře, případně též v /20/, /32/, /34/ aj. se čtenář může přesně seznámit s návrhy různých řadičů přerušení, s časováním přerušení aj.

2.1.6. Přímý přístup do paměti

Na tento pojem jsme rovněž narazili při výkladu činnosti mikroprocesoru. Čtenáře upozorňujeme, že se jedná spíše o speciální činnost, je zde uvedena spíše pro ucelenost, či pro pokročilé uživatele mikropočítače. Při přímém přístupu do paměti (DMA - Direct Memory Access) jsou data z periferie do paměti přenášena bez přímé účasti mikroprocesoru. Přenos dat je řízen speciálním obvodem, tzv. DMA řadičem (pro zajímavost je tento obvod daleko složitější než mikroprocesor). Z rozboru základního algoritmu mikroprocesoru 8Ø8Ø (viz.obr.2.3.) víme, že se požadavek na DMA přenos testuje v době T2, tedy uprostřed strojové instrukce. Obdrží-li DMA řadič žádost o přenos dat, požádá mikroprocesor, (který byl dosud jediným řídičem nad systémovými sběrnicemi) o přidělení těchto sběrnic. Po potvrzení požadavku (mikroprocesor převede výstupy svých sběrnic do třetího stavu a dá o tom zprávu) provede DMA řadič příslušný přenos dat. Po skončení přenosu zruší požadavek na společnou sběrnici a oznámí skončení přenosu mikroprocesoru. Mikroprocesor poté obnoví svoji normální činnost. Vážnější uživatele odkazuji opět na literaturu /30/, /34/, /16/ aj.

Až dosud jsme se v kapitole zabývali některými základními pojmy, se kterými by se uživatel mikropočítače mohl seznámit při studiu architektury mikropočítače. V dalších kapitolách bychom chtěli čtenářům přiblížit programové vybavení mikropočítačů.

3. Programové vybavení mikropočítače

Pod pojem programové vybavení (Software) zahrnujeme programy, které jsou aplikovatelné na daném technickém řešení (Hardware) mikropočítače. Též se vyskytuje termín pro mikroprogramové vybavení (Firmware), které je uloženo v pevných pamětech a podmiňuje jak některé technické tak i programové činnosti mikropočítače.

3.1. Komunikace s mikropočítačem

Sice ještě nežijeme ve století, kdy se s počítačem běžně komunikuje hlasem, ale jsme na tom s klávesnicemi, či jinými vstupy (světelné pero, myš aj.) daleko lépe než naši tvůrci či prvouživatelé mikropočítačů.

V předchozích kapitolách jsme se seznámili se základní architekturou a se základy činnosti mikropočítače a mikroprocesoru. Víme, že mikroprocesor má ve své architektuře zabudován instrukční soubor, který je z hlediska uživatele neměnný (např. u typu 8Ø8Ø). Půjdeme-li ještě na nižší úroveň realizace strojové instrukce, pak je tato instrukce vytvořena mikroprogramem, což je posloupnost mikroinstrukcí. A dále každá mikroinstrukce je realizována jistou posloupností z množiny elementárních funkcí mikroprocesoru. Mikroprogram je např. u mikroprocesoru 8Ø8Ø uživateli nedostupný, je to spíše záležitost pro specialisty, kteří navrhují mikroprocesory.

Vyšším stupínkem než mikroprogram se jeví uživateli již

přístupný strojový kód. Instrukce strojového kódu je vyjádřena jedním až třemi čísly (byty). První číslo udává operační znak, další čísla jsou tzv. operandy, či adresy operandů. Strojový kód představuje uzavřenou množinu operací, které mají v průběhu svého zpracování vždy stejný charakter. Psát programy ve strojovém kódu je sice možné, ale pracné. Základy programování ve strojovém kódu pro IQ 151 jsou popsány např. v /15/. Ještě si můžeme připomenout, že strojový kód je výsledkem činnosti překladačů z vyšších programovacích jazyků typu kompilátor (vysvětleno později).

Příklad: 21 ØØ 1Ø

znamená naplň párový registr HL obsahem 1ØØØH .

Jazyk symbolických adres (nesprávně assembler) je další formou komunikace s mikropočítačem. Je to strojově orientovaný jazyk, umožňuje pružně a beze zbytku využít technické prostředky mikropočítače. Na rozdíl od strojového kódu se instrukce i operandy zapisují symbolicky

Příklad: LXI H, 1ØØØH

Program napsaný v jazyce symbolických adres je v tzv. zdrojovém tvaru. Převod do strojového kódu, což je cílový tvar, provede program, který se nazývá Assembler. Nevýhodou tohoto jazyka je závislost na typu mikroprocesoru.

Ve snaze dále zjednodušit komunikaci s mikropočítačem či s velkými počítači vznikly a dále vznikají nové vyšší programovací jazyky. Bez bližšího rozdělení jmenujme: ADA, ALGOL, APL, BASIC, BCPL, COBOL, CORAL, FORTH, FORTRAN, LIPS, MPL, PASCAL, PILOT, PL/M, PL2, RTL/2 .

Pro vědecko technické výpočty se používají FORTRAN a ALGOL . Umožňují velkou přesnost, jsou prozatím doménou spíše pro velké

počítače. BASIC, známý naopak z mikropočítačů, pracuje interaktivně, je snadno pochopitelný, většina jeho verší je interpret (vysvětleno později), má vlastní editovací a odlaďovací prostředky. Velkou nevýhodou je neexistence normy definice jazyka, existuje mnoho verší jazyka BASIC; programové vybavení v jazyce BASIC není plně přenosné. O BASICu daleko více v /38/. COBOL a RPG se používají na hromadné zpracování dat. Pro programování úloh v reálném čase jsou vhodné CORAL, FORTH, RTL/2 . PASCAL je v současné době na výslunní i u mikropočítačů. Umožňuje vývoj dobře strukturovaných a logicky uspořádaných programů. ADA - perspektivní jazyk, jeho výstavba dovoluje též vytvářet strukturované programy. Je vytvořen z malého počtu struktur a souboru systematických pravidel pro jejich kombinování. Více k jednotlivým vyšším programovacím jazykům speciální literatura pro jednotlivé jazyky a jejich verse.

3.1.1. Realizace programu v mikropočítači

Předpokládejme, že počítač bude zpracovávat program v jazyce BASIC. Nově vytvořený program zavádíme do mikropočítače nejčastěji přes klávesnici. Tento tzv. zdrojový program zpracovává mikropočítač ve dvou krocích. Nejdříve tento zdrojový program přeloží do vnitřního kódu nebo do strojového kódu a poté takto přeložený cílový program provádí.

V zásadě existují dvě odlišné koncepce BASICu, ze kterých vyplývají kompilační a interpretační způsob zpracování programu. Při kompilačním způsobu realizace programu zpracovává mikropočítač zdrojový program v jazyce BASIC ve třech etapách. Nejdříve se tento program přeloží tzv. překladačem (kompilátorem) do strojového kódu, čímž vznikne cílový program. V další etapě je možné tento cílový program spojit s některými dalšími cílovými programy.

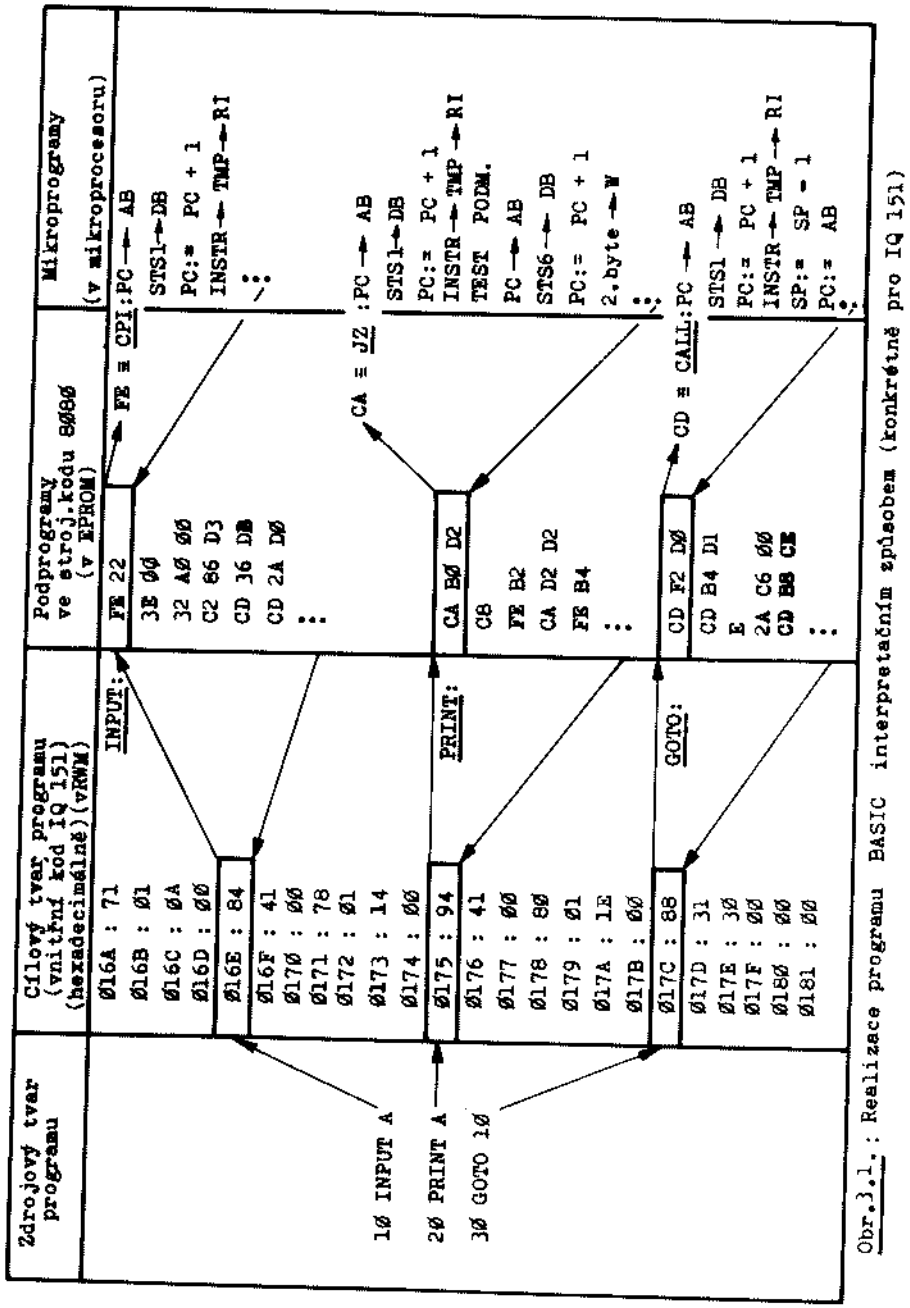
Ve třetí etapě se zavede vytvořený cílový program do paměti mikropočítače a provádí se. Při překladu odhalí kompilátor většinu chyb. Případné chyby se však musí provádět v původním zdrojovém programu (!). (nevýhoda). Program zpracovaný kompilátorem pracuje obecně rychleji než program zpracovaný interpretem, zabírá v paměti menší prostor.

Při interpretačním způsobu realizace programu zpracovává mikropočítač původní zdrojový text ve dvou na sebe navazujících etapách. V první etapě ukládání programu analyzuje překladač (tzv. interpret) jednotlivé řádky zdrojového programu a přeloží je do speciálního vnitřního kódu mikropočítače. Interpret podá uživateli zprávu o případných syntaktických chybách (platí pouze pro některé BASIC-interprety). Druhé etapa začne až po zpracování příkazu RUN. Interpret nejdříve provede např. vymezení oblasti paměti pro data, zpracuje definice uživatelských funkcí atd. Potom začne zpracovávat program ve zprostředkovaném vnitřním kódu. Interpretování vnitřního kódu se realizuje vyvolávením vnitřních podprogramů interpretu. Interpret též ve fázi prvádění zjišťuje chyby. Interpretační překladač umožňuje interaktivní způsob práce s mikropočítačem. Uživatel může snadno modifikovat, či opravovat zdrojový program, může přerušit či pozastavit běh programu, má možnost vypsání programu ve zdrojovém tvaru aj. Při interpretačním způsobu realizace programu musí být interpret stále v paměti (může být v RWM paměti - např. PMD-85, či v ROM (EPROM) paměti - - např. IQ 151), tím však trvale blokuje uživatelskou paměť, dále je jeho nevýhodou opakovaný překlad každého příkazu.

Pokusíme se ještě shrnout celou realizaci programu BASIC a to až do nejelementárnější mikrostruktury v počítači. Předpokládejme, že máme mikropočítač s interpretem BASIC (v pevných pamětech EPROM), mikroprocesor 8Ø8Ø. Konkrétní obr.3.1. platí pro IQ 151. Víme již, že zdrojový program se překladačem jazyka přeloží do speciálního vnitřního kódu. Některé počítače již v této fázi odhalí některé syntaktické chyby, jiné je odhalují až po spuštění programu. Spustíme-li tedy náš program (příkazem RUN), začne vlastní interpretace programu. Interpret BASIC je uložen v pamětech EPROM a skládá se z množství podprogramů ve strojovém kódu, které odpovídají příkazům, funkcím aj. jazyka BASIC. Program se provádí podle čísel programových řádků. Jakmile se narazí na vnitřní kód klíčového slova jazyka BASIC , přenese se provedení tohoto klíčového slova do paměti EPROM, kde se vykoná příslušný strojový podprogram (resp. několik strojových podprogramů). Po ukončení tohoto podprogramu se provádění programu navrátí zpět do hlavního programu BASIC (ten je již samozřejmě v paměti RWM). Opět při zjištění kódu odpovídajícímu klíčovému slovu jazyka BASIC se činnost přenese do strojových podprogramů (v paměti EPROM) atd.

Toto shrnutí realizace BASIC programu v mikropočítači je pro začátečníka zřejmě postačující. Všem uživatelům (i začátečníkům) doporučujeme ještě alespoň globálně se seznámit, jakým způsobem se přiřazuje klíčovému slovu jazyka BASIC příslušný podprogram ve strojovém kódu (tabulka slov, tabulka odskokových adres), (konkrétní ukázky v kap.4.2. při výkladu činnosti MONITORU a BASIC-6).

Pro zvídavější čtenáře se pokusíme ještě naznačit, jakým způsobem se v mikropočítači realizují strojové instrukce.



Obr.3.1.: Realizace programu BASIC interpretačním způsobem (konkrétně pro IQ 151)

3.1.2. Realizace instrukce v mikroprocesoru

Víme tedy již, že se realizace programu BASIC provádí příslušnými podprogramy ve strojovém kódu. Tyto podprogramy jsou tvořeny posloupností instrukcí z instrukčního souboru příslušného mikroprocesoru. Každá instrukce je dále realizována mikroprogramem, který se skládá z mikroinstrukcí. Tato realizace se již provádí v mikroprocesoru (!). A nakonec pro úplně největší „zvědavce" dodejme, že mikroinstrukce se realizuje technickými (hardwarovými) prostředky jistou množinou elementárních funkcí v mikroprocesoru. Avšak tyto podrobnosti už znají jenom vývojoví pracovníci výrobců mikroprocesorů.

Zvídavějším čtenářům se nyní ještě pokusíme rozebrat strojovou instrukci CPI I z obr.3.2. Zároveň se na konkrétním příkladě ozřejmí činnost mikroprocesoru v jednotlivých taktech (M) a dobách (T). Ověřujte si též na obr. 2.2. a na obr. 2.3.

Během prvních tří dob T1 až T3 mikroprocesor přečte operační znak instrukce (FE) adresovaný čítačem instrukcí (PC). Mikroprocesor dává zprávu o provádění činnosti přiložením stavového slova (jedním z deseti stavových slov) STS 1 na datovou sběrnici. V prvním taktu M1 je to zásadně čtení operačního znaku instrukce. Potom PC zvětší (inkrementuje) o jedničku (PC:=PC+1) a dále uloží operační znak do registru instrukce (RI). Během doby T4 uloží původní obsah akumulátoru do pomocného registru ACT. Během druhého taktu M2 mikroprocesor načte přímý operand I do pomocného registru TMP. Tentokráte na datovou sběrnici přiloží jiné stavové slovo STS 6 (čtení vstupního portu) a opět se zvětší o jedničku PC (PC:=PC+1). A nakonec ve třetím taktu M3 se provede porovnání původního obsahu akumulátoru (nyní již obsahu ACT registru) s obsahem registru TMP. Porovnání se provede odečtením v aritmetické-

-logické jednotce, která nastaví příslušné příznaky CY a Z v registru příznaků F.

Projděme si ještě vykonání této instrukce ve stavovém diagramu mikroprocesoru z obr.2.3. Hlavní větví - T1 až T5 - proběhne tato instrukce třikrát než je instrukce skončena. Připomeňme ještě, že v každé době T2 mikroprocesor testuje vstupní signály READY a HOLD a zda-li nemá vykonat instrukci HLT. Podrobnější vysvětlení naleznou čtenáři v /16 / a to i pro ostatní instrukce ze souboru mikroprocesoru 8080.

Viděli jsme tedy, že realizace programu BASIC (interpret) se postupně přenáší z paměti RWM, kde je uložen kódovaný záznam zdrojového programu, do paměti EPROM a v konečné fázi až do mikroprocesoru. Další shrnující pohled na realizaci programu BASIC by nám měl připomenout, že příkazy aj. vyššího programovacího jazyka BASIC se postupně realizují na nižším a nižším programovém vybavení (BASIC → strojový kód → mikroprogram) až se nakonec realizují technickými prostředky v mikroprocesoru.

Kapitoly 1 až 3 jsou pojaty obecně. Čtenáři se seznámili s množstvím nových pojmů. Uživatel začátečník ba ani středně pokročilý uživatel nemusí tuto problematiku podrobně znát. Měl by mít ale podvědomý „pocit", že to či ono pracuje asi „takto".

Další kapitoly jsou už věnované konkrétnímu mikropočítači - IQ 151.

CPI I

Instrukce CPI I porovná obsah střadače A s druhým bytem instrukce (I) a podle výsledku porovnání nastaví příznakové bity Z a CY v registru F. Porovnání se provádí odečtením A - I

CPI I :

M1	T1	PC → AB STS1 → DB
	T2	PC : = PC + 1 INSTR → TMP → RI
	T3	⟨A⟩ → ACT
M2	T1	PC → AB STS6 → DB
	T2	PC: = PC + 1 I → TMP (začátek)
	T3	I → TMP (dokončení)
M3	T1	(začátek porovnání)
	T2	⟨ACT⟩ - ⟨TMP⟩ nastavení F (dokončení)

Obr.3.2. Ukázka realizace strojové instrukce CPI I v jednotlivých taktech a dobách činnosti mikroprocesoru 8080.

Pozn.: 1. Pomocný registr ACT není na obr. 2.2. zakreslen, chápejte ho jako součást akumulátoru A.

2. ⟨A⟩ značí obsah akumulátoru A

4. Základní poznatky o konstrukci a činnosti mikropočítače IQ 151

Tato kapitola byla vlastně důvodem vzniku této publikace. Všeobecné „mikropočítačové beletrie" bylo již napsáno relativně dosti - jmenujme např. /16/, /17/, /12/, /13/, /18/, /19/, /20/, případně některé časopisecké formy /3/, /21/ aj. (i autor této publikace si dovolil v předchozích kapitolách uvést některé základy, ale to spíše ze snahy po ucelenosti publikace). Avšak konkrétní popisy k jednotlivým mikropočítačům chybí.

V této kapitole bychom tedy chtěli uživatelům IQ 151 přiblížit konstrukční technické řešení (hardware) a programové vybavení (software) tohoto mikropočítače. Tato kapitola se může jevit začátečníkům jako obtížnější a zbytečně v některých technických řešeních detailní; nechť si však uživatel z těchto kapitol vybere jen ty poznatky, které ho zajímají.

4.1. Technické řešení mikropočítače IQ 151

Mikropočítač IQ 151 je určen pro školy a to právě jenom pro školy, což je velká výhoda v tom, že ho dostanou pouze školy. Tomu je podřízena i konstrukce ve stolním provedení s masívním krytem, s vestavěným zdrojem atd. Uživatelská dokumentace pro začátečníky je dodávána s mikropočítačem IQ 151 /23/, /24/ a /25/ a jistě ji už všichni uživatelé IQ 151 znají. Jestliže ne, pak si musí být vědomi, že tato publikace předpokládá znalost IQ 151 v rozsahu předchozích příruček.

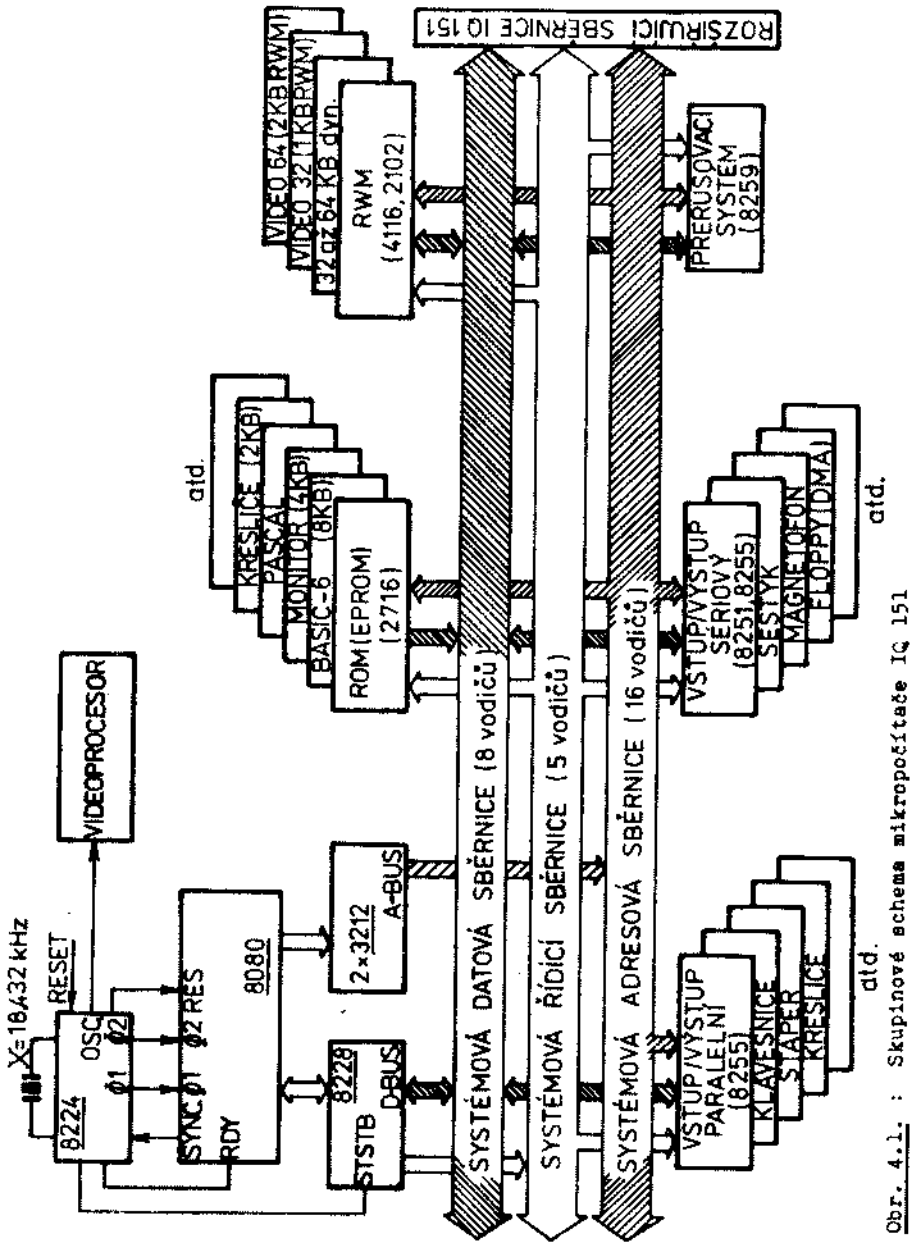
Mikropočítač IQ 151 v základní sestavě s modulem BASIC-6 a VIDEO-32 charakterizuje:

mikroprocesor MHB 8080
paměť RWM 32KB
paměť ROM 8KB BASIC + 4KB MONITOR
VIDEO RAM 1KB
semigrafika 64x64 bodů
membránová klávesnice
obrazovkový displej - s TV přijímačem či monitorem
vnější paměť - magnetofon
rozšiřitelná sběrnice ⇒ otevřený systém
přerušovací systém

Těchto několik parametrů již determinuje IQ 151. Dosti jasně a přehledně si z těchto několika charakteristik udělá uživatel, který se již seznámil s některými jinými počítači, ať prakticky či teoreticky, úsudek o možnostech tohoto mikropočítače, protože jak časem poznáme, jsou i základy činnosti různých mikropočítačů stejné. A jsou převážně určeny technickým řešením kolem mikroprocesoru a základním programovým vybavením. Nyní tedy blíže k technickému vybavení IQ 151.

Skupinové schema IQ 151 je na obr.4.1. Není třeba si ho pamatovat hned na poprvé. Rozebereme si ho po blocích. Hlavní části mikropočítače IQ 151 jsou centrální procesorové jednotka (zahrnující skupinu obvodů kolem mikroprocesoru), paměť RWM, paměť ROM, vstupní a výstupní jednotky. Systémová adresová sběrnice, systémové řídící sběrnice jsou hlavní „tepny", kterými jsou propojeny všechny části mikropočítače. Těmito sběrnicemi proudí informace. Některé jsou jednosměrné, některé obousměrné (vyznačeno šipkami). Nyní se blíže vrátíme k jednotlivým blokům.

Mikropočítač IQ 151 je vybudován na mikroprocesoru MHB 8080 (s tímto mikroprocesorem jsme se již seznámili v kap.2.1.1.). Podpůrné obvody kolem mikroprocesoru jsou generátor dvoufázových hodin (MHB 8224), který užívá jako vnější rezonátor krystal o



Obr. 4.1. : Skupinové schema mikropočítače IQ 151

frekvenci 18.432 kHz. Obvod MHB 8224 dělí tento kmitočet devíti, tedy vlastní kmitočet hodin v IQ 151 je asi 2 MHz. K tomuto obvodu se váže i červené tlačítko - systémový RESET. Dalším z podpůrných obvodů kolem mikroprocesoru je tzv. systémový řadič (MHB 8228), který vytváří a posiluje řídící sběrnici, která obsahuje signály pro paměť - MR* (Memory Read) a MW* (Memory Write), signály pro vstupně výstupní jednotky - IOR* (Input/Output Read) a IOW* (Input/Output Write). Hvězdička u označení znamená, že jsou tyto signály v negativní logice (jsou aktivní do „Ø"). Tento systémový řadič navíc posiluje systémovou datovou sběrnici, která má datové vodiče DØ, D1, D2, D3, D4, D5, D6, D7. Systémová adresová sběrnice má těchto 16 vodičů: AØ, A1, A2, A3, A4, A5, A6, A7, A8, A9, AlØ, A11, A12, A13, A14, A15. Tato adresová sběrnice je rovněž posílena dvěma obvody MH 3212. Tato trojice (či pětice) obvodů (8Ø8Ø + 8224 + 8228 + (2x 3212)) je charakteristická pro téměř všechny mikropočítače užívající mikroprocesor 8Ø8Ø. (V literatuře se často tyto obvody označují pouze čísly a nikoliv i s písmenovým označením, neboť tyto ekvivalenty pod stejným číselným označením vyrábí mnoho světových výrobců). Probrali jsme si tedy technické řešení centrální procesorové jednotky.

Paměť ROM je v mikropočítači užita v mnoha částech a modulech. V IQ 151 se však namísto pamětí ROM (programovaných od výrobce) užívá mazatelných pamětí EPROM. Např. modul BASIC 6 obsahuje 4 ks pamětí EPROM 2716 (sov.ekvivalent PΦ5), která má kapacitu 2KB; tedy celkově 8KB.

Základní programové vybavení IQ 151 - MONITOR - má 4KB paměti ROM (EPROM). Dále např. kresliče Minigraf Ø5Ø7 či AY 413Ø obsahují 2KB paměti ROM (EPROM). Rovněž tak i další rozšiřující moduly jako např. BASIC G, či PASCAL, či ASSEMBLER aj. budou obsahovat pa-

měti ROM (EPROM). Na obr.4.1. si všimněte, že tok dat z pamětí ROM je pouze jedním směrem. Abychom mohli vybrat příslušnou buňku (byte) na kartě ROM je nutno přivést k těmto pamětem adresovou sběrnici, která nám jednak vybíré příslušný obvod (např. jeden z mnoha obvodů ROM) a jednak dále až příslušný byte. Z řídící sběrnice je to pak signál čtení, a to buď paměťového prostoru mikropočítače (např. BASIC-6) signálem MR*, či vstupně výstupní jednotky signálem IOR*. O umístění pamětí ROM v adresním prostoru viz. kap.4.2.

Paměť RWM má v základní sestavě mikropočítače kapacitu 32KB. Jedná se o dynamickou paměť RWM vytvořenou z obvodů MHB 4116, které mají organizaci 16K x 1 bit. Proto musí být vhodným způsobem propojeny, aby vytvořily kapacitu paměti 32K x 8bitů (32KB). Na základní desce v mikropočítači IQ 151 jsou ještě neosazené pozice pro dalších 32KB. Pouhým doosazením obvody MHB 4116 lze rozšířit paměť RWM IQ 151 až na 58KB. (4KB zůstanou pro MONITOR a 2KB pro VIDEO RAM). Inicializační program si hledá vrch RWM paměti, takže IQ 151 pozná, že má větší kapacitu paměti.

VIDEO RAM je rovněž paměť RWM. Má velikost 1KB (32 řádků po 32 znacích). Je vytvořena ze statických pamětí MHB 2102, které mají organizaci 1K x 1 bit, tedy v modulu VIDEO 32 je jich použito 8 kusů. V modulu VIDEO 32 je umístěn ještě tzv. znakový generátor, kde jsou naprogramovány všechny znaky, které užívá IQ 151 včetně semigrafických znaků. Znakový generátor je paměť typu EPROM.

Do pamětí RWM musíme jak zapisovat, tak i z nich číst informace, proto je datové sběrnice k RWM oboustranná (viz obr.4.1.). S adresací je to shodné jako s pamětmi ROM, z řídící sběrnice se využijí signály MR*, MW*, případně pro paměť RWM připojenou

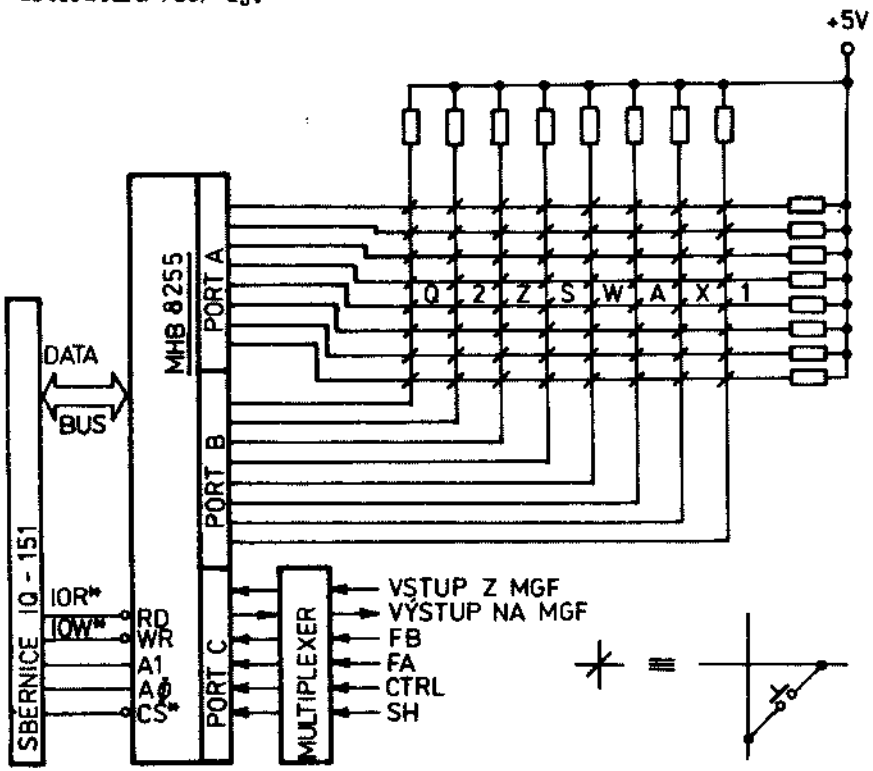
k počítači IQ 151 jako port (např. modul GRAFIK) pak ještě signály IOR* a IOW* (hvězdička opět znamená negativní logiku). O rozložení pamětí RWM v adresním prostoru opět v kap.4.2.

Vstupně výstupní jednotky. Aniž bychom si to hned uvědomovali, je i klávesnice vstupním zařízením mikropočítače. Je připojena jako port mikropočítače. Užitá klávesnice je nejjednodušší-ho provedení. V matici je zapojeno 8 vodorovných a 8 svislých „vodičů". V každém překřížení je umístěno jedno tlačítko znaků (černá tlačítka). Tlačítka FB, FA, SH, CTRL jsou mimo matici tlačítek. Při stisku tlačítka v matici se propojí právě jeden vodorovný a jeden svislý vodič. Na obr.4.2. je zapojení klávesnice. Informace o stisku určitého tlačítka se objeví jak na portu A, tak na portu B integrovaného obvodu MHB 8255. Tento integrovaný obvod patří do rodiny tzv. programovatelných obvodů, obsahuje 3 vstupní nebo výstupní porty (podle naprogramování). Jeho popis je poměrně složitý a proto pokročilé odkazuji na literaturu /26/ aj. Klávesnice je tedy připojena přes porty A a B integrovaného obvodu 8255 (jeden je naprogramován jako vstupní a jeden jako výstupní port). C port je využit pro připojení magnetofonu, některé bity portu C jsou výstupní (výstup na magnetofon) a některé bity jsou vstupní (vstup z magnetofonu, či některá tlačítka klávesnice). V matici je zapojeno 64 tlačítek.

Jednoduchost technického řešení klávesnice je doplněna složitější programovou obsluhou. Ta bude přiblížena v kap.4.2.1.

Dalším vstupně výstupním celým modulem je STAPER, umožňující připojit k IQ 151 tiskárnu (CONSUL 2113), čtečku (FS 1503) a děrovačku (DT 105 S). Obvodové řešení modulu STAPER je opět založeno na vstupně výstupním obvodu MHB 8255. Port A tohoto integrovaného obvodu je použit k připojení čtečky, port B k připojení

tiskárny a děrovače. Některé bity portu C jsou využity pro korespondenční řízení těchto periferií (tj. takové předávání informací, kdy např. periferie potvrdí zpracování znaku a počítač oznamuje platnost přiložených dat). Podrobněji o tomto styku raději opět literatura /26/ aj.



Obr. 4.2. : Zjednodušené zapojení klávesnice mikropočítače IQ 151

Magnetofon jako vstupní a výstupní periferie mikropočítače IQ 151. Magnetofon je užit jako trvalá vnější paměť mikropočítače. Magnetofon je s IQ 151 připojen přes sériový kanál. To je takový styk, kde se informace přenášejí po jednom signálním vodiči (zemní vodič je samozřejmý). Informace o osmi bitech se přenášejí za sebou. (Na rozdíl od paralelního kanálu, kde se informace o osmi bitech přenášejí paralelně po osmi signálních vodičích.) Na rozdíl od jiných mikropočítačů (např. PMD-85) není u IQ 151 užito programovatelného obvodu pro sériový styk (MHB 8251), ale sériový styk je zajištěn programovým vybavením. Je vám jistě známo, že při nahrávání v BASICu příkazem MLOAD se vám program zobrazuje na obrazovce. (Trochu předběhneme - např. klíčové slovo PRINT je na magnetofonu zaznamenáno znakově jako posloupnost bytů písmen P, R, I, N, T a nikoliv jako jednobytový interní kód příkazu PRINT). Rovněž tak nahrávání v MONITORU příkazem L . I zde se nahrávaný program zobrazuje. Tyto způsoby nahrávání zpomalují nahrávku, avšak uživateli poskytují pohodlnou kontrolu, případně kompilaci programů aj.

Ještě několik poznámek k připojení magnetofonu. Vstup č.1 umožňuje nahrávání i přehrávání. Vstup č.2 má zapojen pouze výstup - zde je vidět příprava na možnost zaznamenání např. vypočtených dat na druhý magnetofon. IQ 151 je již připraven pro elektronické zapínání a vypínání magnetofonů. (Pro odvážné a pokročilé jsou to dva tranzistory KF507 kolem MBA810 , kterým lze do kolektorového obvodu zapojit toto ovládání při činnosti MLOAD, MSAVE, L i W . V programové obsluze je nutno doplnit ještě čekací smyčky).

Modul SESTYK je rozšiřující modul pro IQ 151 umožňující sériovou komunikaci IQ 151 přes rozhraní V-24. Je založena na programovatelném obvodu USART (MHB 8251). Podrobněji tento modul rozebírat nebudeme. Rovněž tak další chystané rozšiřující moduly pro IQ 151.

Se sběrnicovou strukturou mikropočítače IQ 151 je ještě svázén přerušovací systém IQ 151 založený na programovatelném obvodu

8259 (československý ekvivalent se nevyrábí). Pro představu - tento obvod je složitější než mikroprocesor. Rovněž tak jeho naprogramování je velmi složité. Ovládat činnost tohoto obvodu v mikropočítači IQ 151 doporučuji pouze velmi pokročilým. Odkazuji opět na literaturu /26/. Tento přerušovací systém má osm přerušovacích vstupů s rozlišenou prioritou. Nejnižší tři priority využívá mikropočítač. Nejvyšší z těchto tří priorit je vám známé tlačítko BREAK. Zbylých pět přerušovacích vstupů je využito v některých rozšiřujících modulech, případně jsou k dispozici uživateli. Doporučení - priority obsazujeme směrem k vyšším prioritám. Pár poznámek o programování přerušovacího systému bude ještě učiněno v kap. 4.2.1.

Poměrně samostatnou část mikropočítače tvoří celek nazvaný videoprocesor. Tento blok není s vlastním systémem mikropočítače příliš svázán a k vlastní činnosti mikropočítače není zapotřebí. Slouží k zobrazení na obrazovce TV přijímače či na monitoru. A proto musí videoprocesor generovat úplný televizní signál s kompletní synchronizační směsí. Většina mikropočítačů včetně IQ 151 nepoužívá prokládané půlsnímky, ale dva stejné půlsnímky. Z kmitočtu krystalu (18.432 kHz) v IQ 151 se vhodným dělením a skládáním vytvoří celá synchronizační směs pro zobrazení na TV přijímači. Užíváme-li propojení s TV přijímačem přes anténní zdířku, potom se ještě vzniklý videosignál musí upravit na vysokofrekvenční signál v tzv. modulátoru. IQ 151 „vysílá" signál asi na 12.kanálu. Podrobnějším rozborem videosignálu se nebudeme zabývat, neboť je obdobný jako u činnosti TV přijímače. Zájemce odkazuji na podrobnější literaturu /27/.

Zbývá nám zmínit se pravděpodobně o „nejchytřejším konstrukčním kousku" a tím je start inicializace na adrese F8ØØH a nikoliv na adrese ØØØØH (což je obvyklé po RESET). To je umožněno

jedním portem (adresa 8ØH) realizovaným obvodem MH 3212. Pro maximální paměťovou kapacitu 58KB RWM by se některé ROM paměti překrývali s RWM. Aby nedocházelo k překrývání paměti RWM např. pamětmi ROM (EPROM) z přídavných modulů, „vysílá" každá takováto paměť signál RAM* do stejnojmenně označeného vodiče na sběrnici IQ 151. V současné sestavě IQ 151 (r.1986) jsou všechny paměti RWM a ROM voleny tak, že se „nepřekrývají". O programové obaluze tohoto portu bude ještě zmínka v kap.4.2.1.

4.1.1. Uživatelská sběrnice mikropočítače IQ 151

A na závěr nám zbývá popsat rozšiřující sběrnici IQ 151. V tab.4.1. jsou uvedeny čísla vývodů na konektoru FRB, označení signálu v zapojení IQ 151, dále pak směr přenosu signálu, dále typ budiče a nakonec vysvětlující poznámka.

Napájení mikropočítače IQ 151 je vícehladinové, což je dáno mikroprocesorem MHB 8Ø8Ø A a pamětmi MHB 4116. Zdroje jsou pro základní sestavu dostatečně dimenzované. Při použití modulu GRAFIK jsou zdroje (hlavně +5V) na hranici dimenzování. (Pomůže výměna diod v +5V zdroji za výkonnější). Jenom pro doplnění poznámka, že napětí -5V musí být zapnuto jako první a odpojeno jako poslední. Ale o to se nám automaticky stará vestavěný zdroj IQ 151.

AØ až A15 tvoří kompletní systémovou adresovou sběrnici mikropočítače IQ 151. Tyto vodiče jsou sice dostatečně posíleny budičem MH 3212, avšak i tak doporučujeme při vlastních aplikacích užívat obvody s malým vstupním proudem (např. dekodéry MH 32Ø5).

DØ až D7 tvoří kompletní systémovou datovou sběrnici IQ 151. Rovněž tato sběrnice je posílena (budičem MH 8228), případně doporučujeme zajistit zesílení datové sběrnice i na vlastních aplikacích (např. užitím MH 3212). (Přes budič MH 3212 je v IQ 151

Vývod konektor FRB	Signál	Směr přenosu	Typ budiče	Poznámka
1	+12 V			Napájení
2	+12 V			Napájení
3	+ 5 V			Napájení
4	+ 5 V			Napájení
5	Ø V			Zem
6	Ø V			Zem
7	AØ	obousměrný	třístavový	Adresová sběrnice
8	A1	obousměrný	třístavový	Adresová sběrnice
9	A2	obousměrný	třístavový	Adresová sběrnice
1Ø	A3	obousměrný	třístavový	Adresová sběrnice
11	A4	obousměrný	třístavový	Adresová sběrnice
12	A5	obousměrný	třístavový	Adresová sběrnice
13	A6	obousměrný	třístavový	Adresová sběrnice
14	A7	obousměrný	třístavový	Adresová sběrnice
15	A8	obousměrný	třístavový	Adresová sběrnice
16	A9	obousměrný	třístavový	Adresová sběrnice
17	A1Ø	obousměrný	třístavový	Adresová sběrnice
18	A11	obousměrný	třístavový	Adresová sběrnice
19	A12	obousměrný	třístavový	Adresová sběrnice
2Ø	A13	obousměrný	třístavový	Adresová sběrnice
21	A14	obousměrný	třístavový	Adresová sběrnice
22	A15	obousměrný	třístavový	Adresová sběrnice
23	DØ	obousměrný	třístavový	Datová sběrnice
24	D1	obousměrný	třístavový	Datová sběrnice
25	D2	obousměrný	třístavový	Datová sběrnice
26	D3	obousměrný	třístavový	Datová sběrnice
27	D4	obousměrný	třístavový	Datová sběrnice
28	D5	obousměrný	třístavový	Datová sběrnice
29	D6	obousměrný	třístavový	Datová sběrnice
3Ø	D7	obousměrný	třístavový	Datová sběrnice
31	MR*	obousměrný	třístavový	Čtení z paměti
32	MW*	obousměrný	třístavový	Zápis do paměti
33	IOR*	obousměrný	třístavový	Čtení ze vstupní či výstupní periferie
34	IOW*	obousměrný	třístavový	Zápis do výstupní či vstupní periferie
35	NRDY*	vstupní	ot.kolektor	Zastavení činnosti mikroprocesoru (1)

Pokračování:

Vývod konektor FRB	Signál	Směr přenosu	Typ budiče	Poznámka
36	HOLD*	vstupní	ot.kolektor	Žádost o DMA přenos (2)
37	HLDA	výstupní	MH 74ØØ	Potvrzení žádosti o DMA přenos (2)
38	RAM*	vstupní	ot.kolektor	Blokování paměti RAM (RWM) (6)
39	INT Ø*	vstupní	ot.kolektor	Žádost o přerušení (3)
4Ø	INT 1*	vstupní	ot.kolektor	Žádost o přerušení (3)
41	INT 2*	vstupní	ot.kolektor	Žádost o přerušení (3)
42	INT 3*	vstupní	ot.kolektor	Žádost o přerušení (3)
43	INT 4*	vstupní	ot.kolektor	Žádost o přerušení (3)
44	VID*	vstupní(4)	ot.kolektor	Videosignál bez synchronizační směsi (4)
45	OSC	výstup	MH 74Ø4	Oscilátor-kmitočet 18.432 kHz
46	Φ2 TTL	výstup	MH 74ØØ	Hodiny Φ2 mikroprocesoru, TTL-úroveň
47	NF	vstup		Akustická signalizace
48				
49				
5Ø	INIT*	výstup	MH 74Ø5	Inicializace, systémový RESET
51	SS*	obousměrný		Snímkové synchronizace (5)
52	SR*	obousměrný		Řádkové synchronizace (5)
53				
54	ZS*	obousměrný		Snímkové zatemnění (5)
55	ZR*	obousměrný		Řádkové zatemnění (5)
56	DMA*	vstupní	ot.kolektor	Činnost DMA (2)
57	Ø V			Zem
58	Ø V			Zem
59	-5 V			Napájení
6Ø	-5 V			Napájení
61	-12 V			Napájení
62	-12 V			Napájení

Tab.4.1.: Rozpis konektoru sběrnice IQ 151

připojen např. BASIC, MONITOR, paměť RWM, VIDEO RAM atd.)

Signály řídící sběrnice MR*, MW*, IOR*, IOW* jsou signály řídící systémové sběrnice IQ 151. Jsou vytvářeny v obvodu MH 8228 a zřejmě nepotřebují další komentář.

(1) Signál NRDY* slouží k „dynamickému" zastavení činnosti mikroprocesoru. Během aktivního signálu NRDY* si mikroprocesor vkládá čekací takty WAIT . Toho se využije při pomalých vstupních či výstupních periferií. Protože je tento vstup pouze jeden je užito vstupu typu otevřený kolektor, aby mohlo zastavit případně pozastavit činnost procesoru více periferií.

(2) Signál DMA* (Direct Memory Access) se využívá u periferních zařízeních s vysokou přenosovou rychlostí. Při činnosti DMA nejsou data přenášena přes akumulátor (případně jiný pracovní registr) a teprve poté na určené místo v paměti, ale data jsou přenášena přímo pod řízením integrovaného obvodu typu DMA (8257), který při své činnosti převezme řízení nad adresovou, datovou a řídící sběrnicí. Mikroprocesor má při tomto DMA režimu „odpojené" sběrnice (jsou ve třetím stavu). Odpojení adresové sběrnice provede signál HOLD* . Vstup typu otevřený kolektor je užit ze stejných důvodů jako v (1). O tom, že mikroprocesor odpojil sběrnice, dává zprávu signálem HLDA .

(3) INTØ* až INT4* jsou přerušovací vstupy IQ 151. V běžné činnosti IQ 151 jsou „maskovány" tj. jsou při aktivaci nečinné. Bude-li je chtít uživatel využít, musí přerušovací systém tvořený obvodem 8259 přeprogramovat. Všechny přerušovací vstupy jsou zapojeny jako otevřený kolektor z důvodů stejných jako v (1) .

(4) Videosignál VID* je z hlediska základní jednotky IQ 151 vstupní signál. Je to TTL úroveň videosignálu bez synchronizační směsi.

Na sběrnici jsou dva kmitočty, které se mohou dále využít. Je to jednak frekvence oscilátoru 16.432 kHz a jednak frekvence hodin Φ2 (přibližně 2 MHz). Signál Φ 2 TTL má TTL výstupní úroveň. (Vlastní hodiny Φ2 pro mikroprocesor mají amplitudu 9 V).

Vstup NF je vstup pro akustickou signalizaci. Je to vstup na nízkofrekvenční zesilovač MBA 810. Možno využít pro vlastní (hardwarové) akustické signalizace. Též je možno uzemněním vstupu NF vyřadit akustickou signalizaci z činnosti. Případné ztlumení akustické signalizace (např. v mikropočítačových učebnách) je možno provést na základní desce trimrem v okolí MBA 810.

Počáteční inicializace INIT* je v činnosti po dobu alespoň tří period hodinového signálu a to jak při zapnutí mikropočítače, tak při stisku tlačítka RESET . Tento signál inicializuje všechny vnitřní i vnější programovatelné i neprogramovatelné obvody do známého stavu. O programové obsluze RESET ještě též v kap.4.2.2.

(5) Synchronizační směs pro vytvoření videosignálu pro monitor (případně pro TV přijímač) je sestavena ze signálů SS*, SR*, ZS*, ZR*. V systému IQ 151 jsou výstupními signály modulů, např. modulu VIDEO-32. Nejsou konstruovány jako sběrnicové signály typu otevřený kolektor, proto jsou z hlediska uživatele spíše jako výstupní signály úrovně TTL.

(6) O signálu RAM* je zmínka v předchozí kapitole.

Zem je společná pro analogové i logické signály. Svorky 5, 6, 57, 58 jsou propojeny.

Až dosud jsme se zabývali technickými prostředky mikropočítače IQ 151. Některé partie jsou více, některé méně podrobné. Jestliže by uživatel porozuměl probírané problematice technického řešení (či znal ještě více), může se považovat za „dobrého", tj. znalého uživatele mikropočítače IQ 151. Shrnutí základních poznat-

ků o mikropočítači IQ 151 vhodné pro začínající uživatele bude provedeno ještě v kapitole 4.3.

4.2. Programové vybavení mikropočítače IQ 151

Technické řešení počítače je nezbytnost. Avšak jenom technické prostředky (hardware) by nám z počítače nikdy neudělaly to,co mikropočítač je. Technickému řešení musíme „vdechnout" život a tím je programové vybavení (software).

Než přistoupí čtenář ke studiu této konkrétní kapitoly, měl by mít alespoň podvědomě jasno o tom jak mikropočítač zpracovává instrukce, jak se ukládá či zpracovává ať strojový či BASICový program, či jak se pracuje s mikropočítačem IQ 151 v režimu MONITOR atd. Některé tyto základy jsou v kap.2 a kap.3, případně v literatuře /23/, /24/, /25/, /14/, /15/.

Samotný mikroprocesor není schopný žádné funkce - to již víme. Doplníme-li ho všemi podpůrnými obvody, tak není rovněž schopen žádné činnosti. Pouze po startu by programový čítač nastavil na hodnotu ØØØØH (hexadecimálně). Teoreticky by mohl pracovat se svými několika registry (A,F,B,C,D,E,H,L,PC,SP), ale lze si jenom velmi obtížně představit, jak by se tyto registry substituovaly, či jak by se kontroloval jejich obsah. Vidíme, že potřebujeme alespoň minimální interakci s počítačem. Tu nechť zajistí klávesnice se svojí programovou obsluhou a obrazovkový displej se svojí programovou obsluhou. Nechť doplníme kompletní mikroprocesorovou skupinu pamětí typu RWM. Dále budeme potřebovat programovou obsluhu na modifikaci buněk. Abychom si mohli uchovat programy, potřebujeme programové vybavení pro nahrání a přehrání byteové struktury. Budeme samozřejmě chtít program spouštět atd. Vidíme, že se nám množí požadavky, a to chceme pracovat s mikropočítačem na

jedné z nejnižších úrovní-pod strojovým kódem. Soubor těchto základních komunikačních a obslužných programů tvoří základní programové vybavení počítače - tzv. MONITOR.

Protože však práce počítače ve strojovém kódu je velmi vzdálena lidské komunikaci, byly vytvořeny vyšší programovací jazyky. Jedním z nich je i BASIC a jeho verse BASIC-6 v IQ 151. Možnosti vyššího jazyka jsou daleko širší, proto i tento vyšší jazyk je daleko složitější a obsažnější. V maximální možné míře se opírá o programy MONITORU.

U mikropočítače IQ 151 jsou tato dvě základní programová vybavení uložena v pevných pamětech EPROM.

Abychom se orientovali v adresním prostoru IQ 151 (verse 32KB) je na obr.4.3. rozdělení paměti a přiřazení jednotlivých lokací. Pozn. je zahrnut i kreslič, který posouvá uživ. oblast (!)

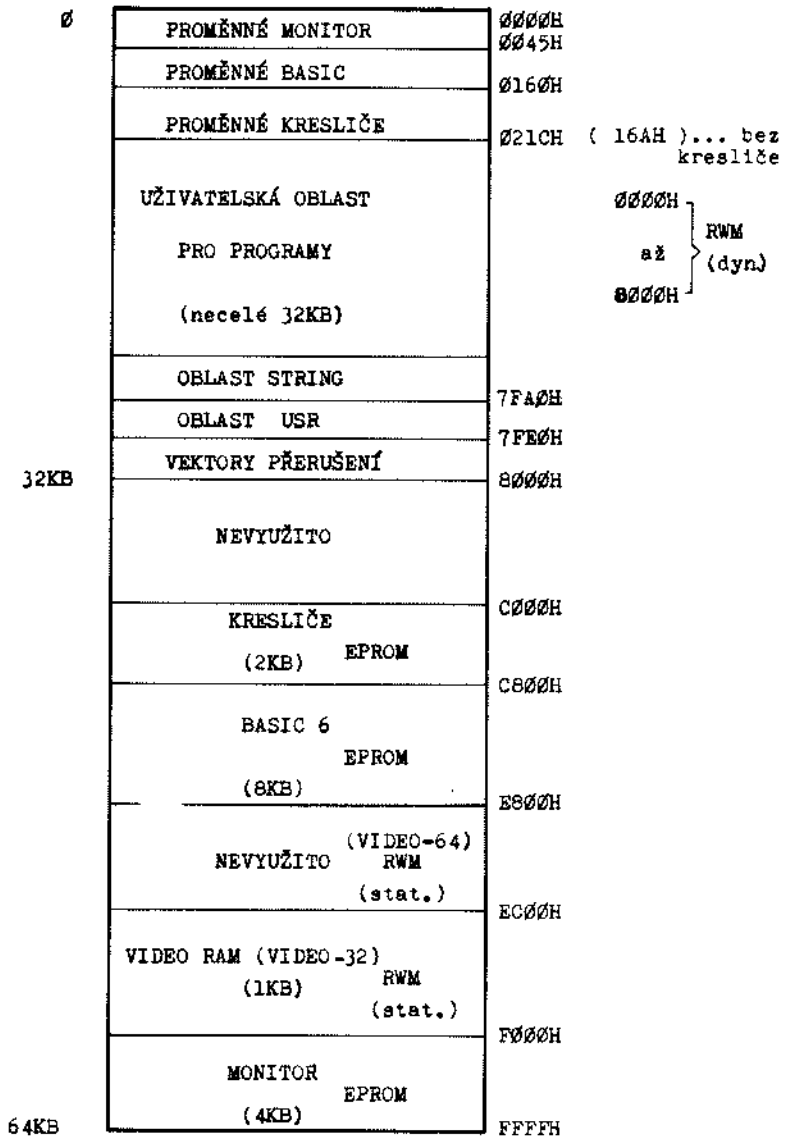
Prozatím si nemusíme pamatovat celé rozdělení paměti, při dalším studiu IQ 151 se budeme k jednotlivým blokům ještě vracet. Obr.4.3. slouží k rychlému orientačnímu přehledu.

4.2.1. MONITOR IQ 151

V této kapitole se pokusíme přiblížit činnost počítače v režimu MONITOR. Bude to pohled z jiné stránky než z uživatelské (ten je rozebrán v /14/), spíše se pokusíme přijít„na kloub věci", či jak to vlastně ten MONITOR pracuje.

Aby nás nesvádělo pracovat s vyšším jazykem BASIC-6, či abychom se přesvědčili, že IQ 151 pracuje i bez modulu BASIC-6, můžeme jej klidně (samozřejmě při vypnutém počítači) vysunout. IQ 151 se po zapnutí přihlásí k činnosti hlášením

```
MONITOR
>□
```



Obr.4.3.: Rozdělení paměti mikropočítače IQ 151

Co se však v mikropočítači děje po zapnutí síťového vypínače? Z rozboru činnosti mikroprocesoru již víme, že po přiložení napětí se programový čítač PC nastaví na hodnotu ØH (ØØØØ hexadecimálně) Jestliže na této adrese nalezne mikroprocesor „rozumnou" instrukci, začne provádět jednotlivé instrukce od adresy ØØØØH postupné za sebou a provede předepsané instrukce a poté se nejspíš zastaví na nějaké čekací či vstupní rutině. Jestliže by na adrese ØØØØH nebyla „rozumná" instrukce, mikroprocesor ji stejně provede a bude pokračovat dále a dále v nekontrolovatelných činnostech. Takový systém bychom nikdy nedostali do definovaného stavu. Mikroprocesorovým systémem, který má MONITOR umístěný od adresy ØØØØH, je např. PMI-8Ø. Po zapnutí tohoto systému se provede inicializace a program se zastaví na rutině vstup z klávesnice.

Ale jak je to s naším IQ 151? Víme, že má od adresy ØØØØH systémové proměnné, a že je to paměť RWM . Či-li po zapnutí počítače je zde nedefinované obsazení. Náš MONITOR je umístěn od adresy FØØØH do FFFFH startovací adresa je F8ØØH . Jak bychom zjistili adresu F8ØØH, kdybychom o systému IQ 151 nic nevěděli? Profesionálně napsané programy (systémové programy) mají některé charakteristiky, na které ještě v průběhu budete upozorněni. Jedním z nich je na začátku systémového programu napsat „odskokovou tabulku" na jednotlivé periferie, se kterými umí mikropočítačový systém komunikovat. Jestliže bychom si nechali vypsat obsah paměti MONITORU buď na obrazovku (příkazem DF8ØØ v režimu MONITOR), případně na tiskárnu, případně provést disassemblaci programu od adresy F8ØØH programem, který se nazývá DISASSEMBLER a který umožňuje překlad strojového kódu do jazyka symbolických adres, objevili bychom právě od adresy F8ØØH odskokovou tabulku:

Adresa(H):	Instrukce(H):			Mnemokód instrukce:
F8ØØ	C3	18	F8	... JMP F818H...start MONITORU
F8Ø3	C3	AA	F8	... JMP F8AAH...vstup z klávesnice
F8Ø6	C3	EB	F6	... JMP F6EBH...vstup ze čtečky
F8Ø9	C3	Ø7	FØ	... JMP FØØ7H...výstup na obrazovku
F8ØC	C3	D9	F6	... JMP F6D9H...výstup na děrovačku
F8ØF	C3	C6	F6	... JMP F6C6H...výstup na tiskárnu
F812	C3	1F	F5	... JMP F51FH...vstup z magnetofonu
F815	C3	6B	F5	... JMP F56BH...výstup na magnetofon

Trochu programování ve strojovém kódu byste již měli umět a mělo by vám být jasno, že tento program, skládající se ze samých nepodmíněných skoků, nic neřeší, že se skutečně jedná o tabulku. K ostatním adresám (F8Ø3H, F8Ø6H, F8Ø9H,...) je připsán jejich význam. Můžeme si ověřit např. výstup na obrazovku. Naplňme registr C např. číslem 41H , což je (ASCII) kód písmene „A" příkazem X-CHANGE :

XC - 41

a proveďme skok na adresu F8Ø9H - výstup na obrazovku, příkazem CALL F8Ø9 (v režimu MONITOR)

CF8Ø9

a na obrazovce se zobrazí písmeno A .
Majitelé tiskárny a děrovačky si ověří, že po naplnění C registru požadovaným znakem a po provedení příkazu CALL F8ØF (výstup na tiskárnu), resp. CALL F8ØC (výstup na děrovačku) se příslušný znak objeví na tiskárně, resp. na děrné pásce. Ostatní odskokové adresy dáváme čtenáři pro úplnou informovanost.

Máme-li v počítači i modul BASIC 6, zkusme přejít do režimu MONITOR (stiskem tlačítka BREAK) a provést příkaz CALL F8ØØ,
(či GO F8ØØ)

CF8ØØ

Po vykonání příkazu se nám na obrazovce objeví hlášení

BASIC

READY

jako kdybychom právě zapnuli počítač. Adrese F8ØØH se říká studený start. Systém mikropočítače IQ 151 přejde do definovaného stavu a vymaže všechny zavedené proměnné. Studený start je ekvivalentní stisknutí RESET (či zapnutí počítače).

Vrátíme se k nadnesenému problému, jak to, že náš počítač má start na adrese F8ØØH a ne na ØØØØH . Elegantním konstrukčním a programovým trikem se dosáhne, že se na adresovou sběrnici připojí (vybere se čip - selektem) nikoliv paměť na adrese ØØØØH , ale „podstrčí" se mikroprocesoru paměť na adrese F8ØØH . (Pro pokročilé a zvlášť zvědavé naznačíme, že programová obsluha tohoto přepnutí je v lokaci F818H až F81AH ; technické řešení: bit DØ portu na adrese 8ØH konstrukčně řešeného obvodem MH 3212).

Mikroprocesor má tedy v programovém čítači adresu F8ØØH a začne provádět jednotlivé instrukce, tak jak jsou postupně zapsány v programu MONITOR.

Jaké budou asi další činnosti našeho startovacího, zaváděcího, či inicializačního systémového programu MONITOR ?

Víme, že náš mikropočítač obsahuje programovatelné obvody 8255 a 8259 a 8255 (pro STAPER) a tyto obvody se musí naprogramovat, inicializovat. Dále si musí program MONITOR „odzkoušet", má-li zasunut např. modul BASIC, či modul kresliče (MINIGRAF Ø5Ø7, XY413Ø), musí umět rozlišit VIDEO 32 a VIDEO 64, musí umět poznat jak velkou má paměť RWM , musí umět obsloužit klávesnici, displej, magnetofon aj., musí umět některé pro uživatele výhodné příkazy (např. GOTO, CALL, FILL, MOVE, CHANGE, LOAD, WRITE, ...), atd., zkrátka musí toho umět relativně hodně. Náš MONITOR

v IQ 151 je pro uživatele velmi bohatý (např. oproti PMD-85). (Pravděpodobně se „zhlédl" ve světoznámém operačním systému ISIS II).

Ještě než přikročíme k „procházce" programem MONITOR je vhodné se zmínit o obsazení portů mikropočítače IQ 151. Porty od ØØH do 7FH jsou ponechány volné pro uživatele. Porty od 8ØH do FFH využívá výrobce IQ 151 a výrobci dalších modulů a periferií. Jestliže bychom o systému neznali naprosto nic, pak se velmi osvědčí program (např. i v jazyce BASIC) vyhledávající určitou bytovou skupinu. Např. budeme-li chtít hledat využité porty, pak budeme hledat byte D3H resp. DBH představující instrukce OUT a IN. Samozřejmě, že dostaneme i některé nesprávné adresy, avšak „rozkoukáním se" v okolí této instrukce určíme, jedná-li se o operaci s portem, či nikoliv (tyto operace jsou např. prováděny přes akumulátor mikroprocesoru).

Takže konkrétně obsazení portů mikropočítače IQ 151:

MHB 8255A	:	port A 84H
(pro klávesnici a magnetofon)		port B ..... 85H port C ..... 86H řídící registr.. 87H
MHB 8255A	:	port A F8H
(pro modul STAPER)		port B ..... F9H port C ..... FAH řídící registr.. FBH
8259	:	..... 88H
(přerušovací systém)		..... 89H
MH 3212	:	..... 8ØH
(start MONITORu od adr. F8ØØH)		

Po zapnutí počítače tedy program MONITOR začne pracovat od adresy F8ØØH . Na adrese F81CH začíná rutina, která zkoumá velikost zabudované paměti RWM . Vždy po 256 bytech se snaží změnit obsah buňky. Jestliže se to podaří, je tam ještě paměť RWM a počítač vyzkouší další buňky o 256 bytů dále. Jestliže se nepodaří změnit obsah buňky, pak již tato pozice není v počítači obsazena pamětmi. Nejvyšší adresa, která šla ještě modifikovat, je vrchol RWM . Tato adresa, jak je vám již z /25/ známo, je v pracovních buňkách MONITORU na adresách ØØØ3H a ØØØ4H .
Po zapnutí počítače IQ 151 verse 32KB je to hodnota

adresa:		obsah
ØØØ3H	:	FFH
ØØØ4H	:	7FH ,

což představuje adresu 7FFFH, a je to právě oněch 32KB .

Nezarazilo vás, že se počítač po zapnutí vždy nastaví do stejného stavu, tj., že některé buňky (např. systémové proměnné v MONITORU - ØØØØH až ØØ2CH jsou vždy stejně obsazeny? To, že se tak pokaždé stane, bude hned samozřejmé. Od adresy F7B1H do adresy F7ECH jsou uloženy konstanty, které se nám při počáteční inicializaci přemístí do našich systémových proměnných v MONITORU. Od adresy F7CDH do F7ECH jsou to odskokové rutiny pro přerušovací systém (podrobněji o přerušovacím systému později). Tato inicializace se v MONITORU provádí hned po najití vrcholu paměti RWM . Další z významnějších činností, které provede inicializační program v MONITORU, je naprogramování programovatelného obvodu MHB 8255A pro STAPER . To si může pilný čtenář ověřit na adresách F899H až F8A4H .

Adresa(H):	Instrukce(H):	Mnemokód instrukce:
F899	3E B4	MVI A, B4H
F89B	D3 FB	OUT FBH
F89D	3E Ø9	MVI A, Ø9H
F89F	D3 F8	OUT FBH
F8A1	3E Ø5	MVI A, Ø5H
F8A3	D3 FB	OUT FB

Pozn.: čtenář, který bude chtít plně pochopit jednotlivé instrukce, se musí nutně seznámit podrobně s činností tohoto obvodu (!) např. v /26/ .

(Pro pokročilé je tento obvod naprogramován v módu 1, A-port je vstup (čtečka), B-port je výstup (tiskárna a děrovačka)). Naprogramování obvodu MHB 8255A pro klávesnici a magnetofon se provádí v podprogramu vstup z klávesnice IQ 151 bez čekání, který začíná na adrese F8C9H a jeho využití si ukážeme ke konci této kapitoly. Inicializace přerušovacího obvodu 8259 začíná na adrese F1B4H . Tento obvod je jeden z nejsložitějších. Velmi vyspělí zájemci o něm naleznou podrobnosti opět v /26/. Pro uživatele stačí pouze to, že má největší prioritu vstup INTØ* a nejnižší INT7* . Odskoková tabulka pro jednotlivá přerušení je umístěna v RWM od adresy 7FEØH do 7FFFH a lze ji tedy modifikovat. Opět pro pokročilé uživatele - všechny uživatelské přerušovací vstupy jsou inicializačním programem zamaskované, tj. nebudou vám reagovat na přerušovací vstupy. Budeme-li chtít využít tyto vstupy (od INT4* do INTØ*) musí se tento obvod 8259 znovu naprogramovat tak, aby tyto vyšší vstupy nebyly zamaskované (!)

Nyní tedy už má MONITOR všechny inicializace za sebou a na adrese F1C7H začíná tento program:

Adresa(H):	Instrukce(H):	Mnemokód instrukce:
F1C7	3A ØØ C8	LDA C8ØØH
F1CA	3C	INR A
F1CB	C2 ØØ C8	JNZ C8ØØH
⋮		

Tento strojový program nám rozliší, máme-li v IQ 151 zasunut modul BASIC 6. Víme, že BASIC 6 začíná na adrese C8ØØH. Tento program tedy načte do akumulátoru obsah buňky na adrese C8ØØH, přičte k němu jedničku a porovná tento obsah akumulátoru s nulou (JNZ adr). Je-li BASIC 6 zasunut, pak je na adrese C8ØØH obsah C3H , pak po inkrementaci je v akumulátoru C4H , což je číslo nenulové, a inicializace MONITORU pokračuje inicializací BASICu. Tu si probereme v následující kapitole. Jestliže by modul BASIC 6 nebyl zasunut, pak je na buňce C8ØØH obsah FFH a po přičtení jedničky je obsah akumulátoru roven nule, tedy skok se neprovede a my zůstaneme v režimu MONITOR. (Malé odbočení: na adrese F851H začíná podprogram, který se ptá na obsah buňky s adresou CØØØH . Toto je první buňka modulu kresliče typu MINIGRAF Ø5Ø7 či XY413Ø. Je-li tato buňka 3EH , potom počítač pozná, že má zasunut modul kresliče, a provede se ještě inicializace kresliče). Počítač bez modulu BASIC 6 tedy vypíše hlášení:

```
MONITOR
>□
```

a čeká na nás v programové smyčce vstup znaku z klávesnice, abychom mu předepsali nějakou činnost v režimu MONITOR. V režimu MONITOR má IQ 151 1Ø příkazů :

```
R ... RETURN
S ... SUBST
C ... CALL
G ... GOTO
L ... LOAD
D ... DISPLAY
X ... CHANGE
W ... WRITE
M ... MOVE
F ... FILL
```

Činnost těchto příkazů by měla být čtenáři již známa (např. /25/ a /14/).

Jsme v režimu MONITOR . Počítač IQ 151 rozumí právě těmto příkazům. Jinak nám otazníkem (?) oznámí, že jsme se dopustili chyby. Seznam těchto 10 „písmen" je jakýmsi „slovníkem" MONITORU . Tato písmena by měla být uložena v počítači, aby si je mohl porovnat a mohl tak rozlišit, jakou činnost od počítače budeme vyžadovat. Každý z příkazu MONITORU představuje jistý strojový program, který realizuje požadovanou činnost. Každý z těchto programů má určenu začáteční adresu. Jestliže bychom prostudovali více mikropočítačů, poznali bychom, že jsou vždy poblíž sebe v programu umístěny jak slova (tj. soubor R,S,C,G, ...), tak odskokové adresy. Pokusíme se toto místo v počítači nalézt.

Vtipně využijeme příkazu MOVE v režimu MONITOR tak, že požadovanou oblast, kterou chceme prozkoumat, přesuneme do lokace VIDEO RAM (tj. od EC00H do EFFFH). Příkaz

MF000,F3FF,EC00

nám na VIDEO obrazovce zobrazí první KB MONITORU . Pokusíme se v této „změti" nalézt náš slovník RSCG...MF . A skutečně se nám to povede hned na poprvé, asi uprostřed v dolní polovině obrazovky nalezneme posloupnost znaků

..... RSCGLDXWMF.....

Odpočítáním adresy zjistíme, že R začíná na adrese F209H . Příkazem DISPLAY opět v režimu MONITOR si to ověříme a navíc objevíme odskokovou tabulku příkazů:

DF209

Adresa (H):	Obsah (H):						
F209 :	52 R	53 S	43 C	47 G	4C L	44 D	58 X
F210 :	57 W	4D M	46 F				

a hned dále následují jistá čísla. Je zřejmé, že tato čísla netvoří strojový program, ani netvoří další znakovou tabulku. Při větších znalostech z tvorby programových systémů bychom mohli prohlásit, že jsme objevili adresy odskoků jednotlivých příkazů MONITORU :

Adresa (H)	Obsah (H)	Příkaz	Adresa příkazu (H)
F213 :	51 F2	FILL	F251
F215 :	40 F2	MOVE	F240
F217 :	60 F2	WRITE	F260
F219 :	C4 F2	CHANGE	F2C4
F21B :	85 F3	DISPLAY	7385
F21D :	B6 F3	LOAD	F3B6
F21F :	05 F4	GOTO	F405
F221 :	FC F3	CALL	F3FC
F223 :	38 F4	SUBST	F438
F225 :	84 F4	RETURN	F484

Přiřazení adres příkazů k jednotlivým příkazům musí mít „logiku". V našem případě prvnímu písmenu R odpovídá poslední adresa.

Zkusme se podrobněji podívat např. na jednoduchý příkaz FILL. Začíná na adrese F251H:

Adresa (H):	Instrukce (H):	Mnemokód instrukce:
F251	04	INR B
F252	CD E5 F4	CALL F4E5H
F255	C1	POP B
F256	D1	POP D
F257	E1	POP H
F258	71	MOV M,C
F259	CD 9B F4	CALL F49BH
F25C	D2 58 F2	JNC F258H
F25F	C9	RET

Delším analyzováním bychom zjistili, že v B registru je standardně 2. Máme-li 3-parametrový příkaz (např. FILL,MOVE,WRITE) je na prvním místě instrukce INR B, máme-li jednoparametrový příkaz (např. DISPLAY) je na prvním místě podprogramu pro příkaz MONITORU instrukce DCR B. V registru B je tedy počet parametrů příkazu. Každý program ve strojovém kódu analyzujeme tak, že procházíme nejdříve hlavní větví a nepouštíme se do analyzování podprogramů. Nechť instrukce CALL F4E5 cosi vykoná. (Prozradíme, že čte naše parametry zadané z klávesnice.) Potom následují tři instrukce „vytahující" ze zásobníku (STACKU) nějaké parametry. Vlastní přesun (naplnění) nastává programem počínaje adresou F258H . Variací s náplní párových registrů BC, DE, HL bychom si ověřili, že v párovém registru BC je plnící znak (resp. pouze v C registru), v pár.registru DE je konečná adresa plnění, v pár.registru HL je počáteční adresa plnění.

Proveďme tento pokus: příkazem CHANGE v režimu MONITOR naplňme registry B,C,D,E,H,L takto:

XB - ØØ
XC - 3Ø ASCII kód „Ø" je 3ØH
XD - EF
XE - FF
XH - EC
XL - ØØ

a proveďme spuštění příkazem CALL F258 :

CF258

Celá obrazovka se nám zaplní nulami. Zvládli jsme tedy programové obsazení paměti znakem.

Příkaz MOVE si rozanalyzujte sami. Víte, že začíná na adrese F24ØH. Prozradíme vám, že v registru BC je adresa určení, v registru DE je koncové adresa bloku, v registru HL je počáteční adresa přesunovaného bloku. Vlastní rutina začíná na adrese

F247H . Obdobně, naplníme-li registry BC, DE, HL (např. příkazem CHANGE) a zavoláme rutinu na adrese F247H příkazem CALL F247:

CF247 ,

provede se požadovaný přesun (programově).

Příkladem jednoparametrového příkazu je DISPLAY . Tento příkaz začíná na adrese F385H . Čtenář nechť si sám rozebere první tři instrukce (DCR B, CALL F4E5, POP H, ...). Instrukce POP H dává tušit, že se právě do párového registru HL bude ukládat vstupní parametr (počáteční adresa). A na následující adrese F38AH bude pravděpodobně začínat vlastní rutina pro DISPLAY.Ověřením zjistíme, že tomu tak skutečně je. Zkusme naplnit registr HL adresou,odkud budeme provádět výpis paměti. Např. opět příkazem CHANGE naplňme HL registr adresou např. F8ØØH

XH - F8
XL - ØØ

a proveďme příkaz CALL F38A . Na obrazovce získáme výpis obsahu paměti od adresy F8ØØH , právě tak,jako bychom provedli příkaz DISPLAY

DF8ØØ

z klávesnice.

Příkaz RETURN v režimu MONITOR začíná na adrese F484H , to již víme. RETURN nemá žádných vstupních parametrů, a proto vyzkoušíme hned počáteční adresu. Zkusme tedy provést příkaz CALL F484 :

CF484

a stane se totéž jako bychom stiskli tlačítko R v režimu MONITOR.

Příkaz SUBST v režimu MONITOR . Po prohlédnutí prvních čtyř instrukcí počínaje od adresy F438H znalejší zjistí (začátečníkům prozradíme), že v párovém registru HL by měla být počá-

teční adresa substituce a vlastní skok do rutiny SUBST je na adrese F441H . Proveďme tedy příkazem CHANGE naplnění HL registru:

XH - 1Ø
XL - ØØ ...(adresa 1ØØØH)

a poté CALL F441 :

CF441

a můžeme již substituovat známým způsobem od zvolené adresy 1ØØØH.

Příkaz GOTO v režimu MONITOR je částí příkazu CALL v režimu MONITOR . Počáteční adresy známe. (Pro GOTO je F4Ø5H a pro CALL je F3FCH.) Provedeme-li například v režimu MONITOR příkaz CALL F3FC :

CF3FC

objeví se blikající kurzor. Při analýze zjistíme, že je to projev podprogramu vstupu parametru z klávesnice. Napíšeme tedy z klávesnice např. známou adresu F8ØØH (start.adresa) a odešleme tlačítkem CR . Na čisté obrazovce se objeví nápis BASIC READY obdobně tak, jako bychom provedli RESET .

Obdobně si vyzkoušíme GOTO . Provedeme příkaz CALL F4Ø5

CF4Ø5

Blikající kurzor nás upozorní, že počítač očekává parametr z klávesnice, vložíme mu tedy z klávesnice např. adresu F4B4H (to je začáteční adresa RETURN) a po odeslání se skutečně dostaví výsledek jako bychom RETURN provedli.

Příkaz WRITE v režimu MONITOR je tříparametrový. Po prohlédnutí si prvních pěti strojových instrukcí počínaje startovací adresou WRITE (F26ØH) a po několika variacích, kam patří jednotlivé vstupní parametry, bychom zjistili, že obsadíme-li párový registr BC startovací adresou, DE koncovou adresou bloku, HL

začáteční adresou bloku, který chceme zaznamenat (obsazení těchto registrů můžeme provést např. příkazem CHANGE v režimu MONITOR) a provedeme-li příkaz CALL F267 :

CF267

Ozve se nám na magnetofon zaznamená požadovaný blok. Záznam se provádí v blocích po 8Ø bytech (= 5ØH), s následující hlavičkou: Např.:

: 5Ø 1ØØØ ØØ CDE7C4 ...

Nejdříve dvojtečka, potom počet bytů v bloku (je-li blok menší než 8Ø znaků, pak je zde menší číslo než 5ØH, 1ØØØ je počáteční adresa, kam se uloží první byte (naše CD v uvedeném příkladě), ØØ je standardně oddělovač. Další bloky jsou zaznamenány ve tvaru

: 5Ø poč.adresa ØØ další byte strojových instrukcí

Koncový záznam má tvar

: 5Ø startovací adresa Ø1 kontrolní součet

Čili z nahrávky lze určit jak její umístění, tak její startovací adresu. Zaznamenaný kontrolní součet se při nahrávce porovná (kontrola).

A zbývá nám ještě příkaz LOAD v režimu MONITOR . Počáteční adresu tohoto příkazu známe (F3B6H). Po rozebrání dvou prvních instrukcí zjistíme, že na vrcholu zásobníku musí být hodnota ØØØØH, chceme-li nahrávat jako by standardním příkazem L (bez parametrů, či LØ). Napíšeme tento krátký program ve strojovém kódu např. od adresy 5ØØØH :

Adresa (H):		Instrukce (H):	Mnemokód instrukce:
5ØØØ	:	21 ØØ ØØ	LXI H, ØØØØH
5ØØ4	:	E5	PUSH H
5ØØ5	:	C3 BA F3	JMP F3BAH

Prvé dvě instrukce nám na vrchol zásobníku nastaví ØØØØH . Další instrukce je již skok do rutiny příkazu LOAD. Principiální ukázku použití předvedeme při načtení libovolného programu v režimu MONITOR . Po napsání uvedeného programu do počítače vložíme ještě příkaz

C5ØØØ

(ale neodešleme ho - tak jako při nahrávání příkazem L). Odeslání provedeme až při zaváděcím (pilotním) kmitočtu nahrávky z magnetofonu.

Seznámili jsme se již s příkazy MONITORU, našli jsme odskokovou tabulku, pohráli jsme si s některými příkazy. Nyní si ještě ukážeme, jak přiřadí počítač požadované odskokové adresy a jak zanese do podprogramu parametry.

Po zapnutí počítače a již zmíněné inicializaci se činnost počítače „dynamicky" zastaví na adrese F1E3H (bez modulu BASIC 6):

F1E3 : CD ØØ FØ CALL FØØØH

Prozradíme, že je to podprogram vstupu znaku z klávesnice (s čekáním). Hlavní podprogram pro přiřazení příkazu je od adresy F1E3H až do adresy F2Ø8H (pilný čtenář si ho jistě vypíše sám). Počítač tedy čeká na vstup z klávesnice. Stiskneme-li nějaké jemu známé slovo (tj. písmeno R,S,C,G,...tedy vlastně příkaz MONITORU) počítač projde pěkně popořádku tento soubor písmen a zároveň se posouvá v odskokové tabulce se začátečními adresami jednotlivých příkazů. Při shodě vloženého znaku z klávesnice s některým z jeho souboru (R,S,C,G,...) se provede skok na danou adresu. A dále každý podprogram začíná vstupem parametrů (opět přes klávesnici). Parametry se nezadávají postupně, ale najednou, oddělené čárkou. Neodpovídá-li vložený znak ani jednomu jeho znaku (R,S,C,G,...), potom na nás počítač pípne a vypíše na obrazovku „?" (otazník),

což je chybové hlášení v režimu MONITOR .

Pokusíme si tuto činnost ukázat na následujícím příkladu. Vložme do počítače instrukci CALL F1E3 :

CF1E3

Počítač nám blikajícím kurzorem oznamuje, že očekává vstupy z klávesnice. Napišme tedy z klávesnice např.

FECØØ,EFFF,3Ø

Všimněte si, že tento příkaz (vlastně pouze vstup z klávesnice) je napsán bez vodícího znaku „ > " v režimu MONITOR (!). Pro méně zkušené - tento příkaz by měl zaplnit nulami (= 3ØH) celou VIDEO RAM (od ECØØH do EFFFH). Zapsanou předcházející sekvenci vstupu z klávesnice odešleme (CR) a skutečně se dostaví vykonání předpokládané instrukce. Analogicky možno ověřit např. též

MØ,4ØØ,ECØØ

čímž se na obrazovce objeví první KB paměti (vysvětlete sami).

Na začátku této kapitoly jsme příkazem v režimu MONITOR

MFØØØ,F3FF,ECØØ ... zobrazili první KB z programového vybavení MONITOR.

Obdobně zobrazme na obrazovce další KB z tohoto programového vybavení.

MF4ØØ,F7FF,ECØØ ... zobrazí na obrazovce druhý KB z MONITORU. Na obrazovce se vám nejspíše nepodaří objevit nic, co by nás navedlo na další „objevování" v MONITORU. Prozradíme, že v této oblasti jsou některé programové obsluhy příkazů MONITORU a dále je zde množství podpůrných podprogramů pro programy příkazů v MONITORU. Jejich rozbor a možnosti jejich využití je nad rámec této publikace.

MF8ØØ,FBFF,ECØØ ... zobrazí na obrazovce třetí KB z MONITORU. Od poloviny obrazovky objevíme „znaky" odpovídající obsahu „FF"

(přesvědčte se příkazy DISPLAY či SUBST). Obsah bytů „FF" znamená u pamětí typu EPROM nenaprogramovanou paměť. Tyto nenaprogramované byte objevíme i v celém čtvrtém KB MONITORU. Přesvědčte se příkazem

MFCØØ,FFFF,ECØØ

Vidíme tedy, že čtvrtý KB a část třetího KB v MONITORU není využita. Zkušení uživatelé si mohou do této nevyužité oblasti doplnit své vlastní uživatelské podprogramy.

Ještě však zůstaneme u třetího KB MONITORU. Objevíme na obrazovce např. sekvenci

1234567QWERTYUIASDFGHJKZXCVBNM...

Zvláště část sekvence QWERTY by nám mohla napovědět, že se bude jednat o tabulku znaků tlačítek z klávesnice. Naše klávesnice v mikropočítači IQ 151 vám nedodá totiž kód odpovídající hodnotě v ASCII tabulce, ale dodá nám pouze informaci o poloze v matici 8x8 tlačítek (viz. technické řešení klávesnice IQ 151). A právě pomocí uvedené tabulky se kód tlačítka z matice 8x8 převede na kód ASCII,kterému již „rozumí" počítač. V okolí této tabulky znaků by pravděpodobně mohl být obslužný podprogram pro klávesnici. Skutečně tomu tak je. Již z odskokové tabulky na startu MONITORU víme, že podprogram obsluhy klávesnice začíná na adrese F8AAH. Zkusme zadat příkaz CALL F8AA v režimu MONITOR :

CF8AA

Vidíme, že blikající kurzor na nás čeká, čeká na náš stisk klávesy. Tento vstup z klávesnice se nazývá vstup z klávesnice s čekáním. Po stisku černé klávesy se na obrazovce objeví opět vodící znak „ > " a kód (ASCII kód) stisknuté klávesy se objeví v akumulátoru mikroprocesoru (A). Stiskneme-li např. „B" objeví se v akumulátoru jeho kód (= 42H), o čemž se můžeme přesvědčit příkazem

CHANGE v režimu MONITOR.

Na adrese F8C9H začíná podprogram vstupu z klávesnice bez čekání. Rozlišení kursových šipek (posuvů) na pravé části IQ 151 přináší začátečníkům jisté potíže, neboť nelze využít příkazu BASICu INKEY$. Ukážeme se praktickou ukázku na rozlišení kursových šipek v BASICu s využitím podprogramu vstupu z klávesnice bez čekání.

```
1Ø  PRINT  USR(HEX(F8C9))
2Ø  GOTO   1Ø
```

Po spuštění tohoto programu se na obrazovce začne tisknout hodnota

138 ... odpovídá nestisknuté klávesnici,
Ø ... odpovídá stisku černé klávesy,
32 ... →
8 ... ←
25 ... ↑
26 ... ↓
12 ... ↖
} stisk odpovídá příslušným kurzorovým šipkám.

Čtenář si jistě udělá programy, které rozliší „hodnotu" příslušných stisknutých tlačítek (zdůrazňuji bílých tlačítek). Připomenutí - funkce USR vám vynáší ze strojového podprogramu obsah akumulátoru.

Budeme-li chtít tímto podprogramem rozlišit černá tlačítka musíme jejich stisk rozlišit v C-registru mikroprocesoru. Napíšeme kratičký program v jazyce BASIC

```
1Ø  PRINT   USR (HEX(6ØØØ))
2Ø  GOTO    1Ø
```

a k němu strojový program začínající na adrese 6ØØØH:

Adresa (H):	Instrukce (H):	Mnemokód instrukce:
6ØØØ :	CD C9 F8	CALL F8C9H*
6ØØ3 :	79	MOV A,C
6ØØ4 :	C9	RET

Po spuštění programu se nám na obrazovce budou tisknout následující hodnoty

Ø ... nestisknutá klávesnice

65 ... stisk A (pozn. 65 = dekadická hodnota ASCII kódu „A")

8 ... stisk ←

atd.

(Poznámka: Těmito programy nelze rozlišit mezeru a → , oba dávají hodnotu 32)

Příklady použití zase necháme na uživateli. Připomeňme, že pro rozlišení černých tlačítek lze využít snadněji funkci INKEY $.

Na závěr rozboru MONITORU se ještě mohou velmi zkušení a pokročilí uživatelé (začátečníci nechť se k této problematice vrátí později) IQ 151 obeznámit alespoň se základními možnostmi přerušovacího systému IQ 151. Tento přerušovací systém je založen na integrovaném obvodu typu 8259 a má 8 přerušovacích vstupů s rozlišenou prioritou. Princip přerušení jsme probrali v kap.2. Připomeňme jenom, že přerušení činnosti mikropočítače vyvolá změna úrovně logického signálu na vstupech INT Ø* až INT 7*. Poté je rozlišena priorita požadavků přerušení a je-li splněna, pak se předá řízení příslušnému podprogramu pomocí strojové instrukce CALL adresa z příslušného řádku přerušovací tabulky. (Např. při INT Ø* pro IQ 151 to bude konkrétně instrukce CALL 7FEØ .) Tři nejnižší priority využívá mikropočítač (INT 7*, INT 6*, INT 5*). INT 5* je vám známé tlačítko BREAK .

Vektory přerušení (tj. vlastně počáteční adresy podprogramů přerušení) jsou uloženy v proměnných MONITORU (paměť RWM) od

adresy 7FEØH takto:

Adresa (H):	Vektor přerušení:	
7FEØ :	C3 ØØ F8 FF	... INT Ø
7FE4 :	C3 ØØ F8 FF	... INT 1
7FE8 :	C3 ØØ F8 FF	... INT 2
7FEC :	C3 ØØ F8 FF	... INT 3
7FFØ :	C3 ØØ F8 FF	... INT 4
7FF4 :	C3 92 F7 FF	... INT 5
7FF8 :	C3 48 F7 FF	... INT 6
7FFC :	C3 ØØ F8 FF	... INT 7

Tato tabulka vektorů se nastaví vždy po inicializaci IQ 151 (tj. po zapnutí, či po RESET). Přerušovací systém je ve stavu přerušení povoleno (instrukce EI), avšak všechny přerušovací vstupy INT Ø* až INT 4* jsou „zamaskované". Tj. nelze je bez programového odmaskování využít. Změnou masky přerušovacích vstupů si ukážeme některá „téměř kouzla" s počítačem. Maska přerušení se posílá na port 89H . Takže vyzkoušejme v přímém režimu BASIC odeslat tyto příkazy :

```
OUT HEX(89),HEX(DF):CALL HEX(CCBA)
```

a zjistíme, že máme zablokovanou klávesnici a že funguje pouze tlačítko BREAK (a samozřejmě RESET).
(Pozn. při stisku klávesnice zobrazí první písmeno, ostatní již nelze z klávesnice zadat. Příkaz CALL HEX(CCBA) je nutný, neboť kdykoliv se při normálním provozu počítače objeví READY nastaví se inicializační parametry a tedy i původní inicializační maska přerušení. Programem od adresy CCBAH vstoupíme do inicializačního programu, ale až za instrukce obnovy inicializační masky.)
Proveďme RESET a odešleme příkaz v přímém módu BASIC :

```
OUT HEX(89),HEX(BF):CALL HEX(CCBA)
```

a zjistíme naopak, že máme „zamaskované" tlačítko BREAK a klávesnici ne. Budeme-li chtít využít uživatelské přerušovací vstupy INT Ø* až INT 4* musíme provést jejich odmaskování (tj. na místo „1" v původní masce dosadit masku s „Ø" pro požadované přerušení). Toto se musí provést obslužným programem ve strojovém kódu. Dále odkazuji uživatele na literaturu /26/.

Ještě si ukážeme jak je možno pracovat s tabulkou přerušovacích vektorů. Substituujme na místo vektoru pro BREAK jiný vektor - např. adresu pro podprogram pípnutí, který začíná na adrese F973H (a lze si to odzkoušet např. příkazem CALL F973 v režimu MONITOR) takto: (příkazem SUBST)

```
S7FF4  :  C3 - C3
 7FF5  :  92 - 73
 7FF6  :  F2 - F9
```

(původní vektor C3 92 F2 jsme změnili na C3 73 F9). Vraťme se do režimu BASIC stiskem „R" , nikoliv RESET - to by se náš substituovaný přerušovací vektor přepsal na původní. Nyní po stisku BREAK počítač pouze pípne, ale BREAK se neprovede. Toto lze provést pouze jednou, neboť podprogram pípnutí si nezrušil svoji masku přerušení a další přerušení nelze tedy provest. Ale to už je opravdu příliš složité a tyto uživatele odkazuji na literaturu /26/. Většina uživatelů si vystačí s několika uvedenými možnostmi. Systemové proměnné MONITORU na adresách ØØØØH až ØØ2DH byly popsány a objasněny již v předchozích publikacích /14/, /25/ , a proto se jimi nebudeme v naší knize zabývat.

Tímto bychom chtěli popis činnosti MONITORU uzavřít. Pro uživatele byla asi náročná, avšak konkrétní. Uživatel nechť si bere z této záplavy informací jen ty, kterým porozuměl a které v dané situaci využije. V žádném případě nebylo jejich smyslem, aby si je zapamatoval. Obecné shrnutí o činnosti MONITORU bude

provedeno v kap. 4.3.

4.2.2. BASIC IQ 151

Přistoupíme dále k přiblížení vyššího programového vybavení IQ 151 - a tím bude programovací jazyk BASIC 6. Opět bude nejlépe, budete-li mít vedle této publikace připravený mikropočítač IQ 151 a hned si všechno pěkně odzkoušíte.

BASIC 6 v mikropočítači IQ 151 je interpretační. Tento pojem jsme si již objasnili v kap. 3. Rozsah BASICu 6 je 8KB a připomeňme si, že je umístěn v adresovém prostoru C8ØØH až E7FFH.

Z rozboru realizace BASIC programu v mikropočítači v kap. 3.1.1. je nám již zřejmé, že se programové vybavení BASIC 6 bude skládat z množství podprogramů ve strojovém kódu mikroprocesoru 8Ø8Ø, které jsou zaznamenány v paměti ROM - EPROM a kterými se realizuje požadovaná činnost.

Uživatelský BASIC program se ukládá do paměti RWM od jisté pevné adresy - od 16AH pro IQ 151 bez kresliče a od 21DH s kresličem - - v kódovaném tvaru, který byl již popsán ve /14/ a čtenář by toto měl již znát (částečně bude tato problematika připomenuta i v této kapitole). Programy se do počítače vkládají přes klávesnici, též je možné nahrát programy z magnetofonu, resp. z děrné pásky, resp. z floppydiskové jednotky. Každý programový řádek BASIC programu začíná číslem řádku. Podle těchto čísel řádků se řídí jak běh programu (po RUN) , tak i případná editace programu. Mikropočítač IQ 151 může pracovat jak v přímém,tak i programovém režimu. V přímém režimu příkazový řádek nezačíná číslem řádku. Po odeslání takového příkazového řádku (tlačítkem CR) se naprogramovaná činnost ihned provede. V programovém režimu příkazový řádek začíná číslem řádku. Po odeslání tohoto řádku se požadovaná činnost neprovede, pouze se tento pro-

gramový řádek jistým způsobem zakóduje do uživatelské části paměti RWM. Teprve po odstartování programu příkazem RUN se vykoná naprogramovaná činnost.

Tak jako MONITOR, potřebuje i BASIC 6 jisté proměnné - systémové proměnné - kde si může mikropočítač dočasně či trvale poznamenat některé své stavy nebo okamžité hodnoty. Tyto systémové proměnné BASIC 6 jsou v mikropočítači IQ 151 uloženy od adresy 45H do 169H, resp. od 45H do 21CH v konfiguraci s kresličem, za systémovými proměnnými MONITORU. Jistě je samozřejmé, že jedná o paměť RWM.

Po zapnutí mikropočítače IQ 151 (máme zasunuty moduly BASIC 6 a VIDEO 32) se na obrazovce objeví BASIC READY. Mikropočítač je v jistém klidovém stavu a očekává od uživatele zadání požadované činnosti přes klávesnici mikropočítače. Vkládané znaky se zobrazují na obrazovce, ale též se ukládají do vstupního zásobníku (bufferu). Zde se rovněž provádějí případné editace programového řádku. Stisknutím tlačítka CR oznámíme mikropočítači konec zadávání, tj. též odeslání programového řádku.

V dalších kapitolách se pokusíme probrat naznačené činnosti programového vybavení BASIC 6 podrobněji a s konkrétními praktickými příklady.

4.2.2.1. Start BASIC 6

V kap. 4.2.1. jsme si ukázali, co se stane po zapnutí mikropočítače, jak proběhne inicializace MONITORU, případně i jiných periferií např. kresliče , jak IQ 151 pozná, které moduly má zasunuté. Po těchto inicializacích se přejde z MONITORU na počáteční adresu BASIC 6 tj. C8ØØH a začne se provádět počáteční inicializace systémových proměnných jazyka BASIC 6. Takovýto start programového vybavení (MONITOR i BASIC 6) je nutný, protože uvede vždy po zapnutí mikropočítače či po RESET celý systém IQ 151 do přesně definovaného stavu. Uvědomte si, že paměť pro uživatelské programy i pro systémové proměnné je typu RWM, které např. po vypnutí mikropočítače definitivně ztratí obsah informace.

Podíváme se tedy na počáteční adresu BASIC 6 např. příkazem SUBST C8ØØH.

Adresa H	:	Instrukce H				:	Mnemokód instrukce :
C8ØØ	:	C3	ØC	C8	...		JMP C8ØCH ... studený start BASIC
C8Ø3	:	C3	7D	DB	...		JMP DB7DH ... výpis na obrazovku, ukazatel HL registr
C8Ø6	:	C3	81	DØ	...		JMP DØ81H ... BREAK READY
C8Ø9	:	FF	FF	FF	...		... není využito

Obdobně jako v programovám vybavení MONITOR je i na prvních adresách odskoková tabulka některých častěji používaných rutin.

První odskok na adrese C8ØØH se nazývá studený start BASIC 6, který provede úplnou počáteční inicializaci proměnných BASIC 6, zruší (vynuluje) všechny vypočtené jednoduché i indexované proměnné, případně též zruší uživatelský BASIC program. Systémové proměnné MONITORU však nemění. Ověřte si tuto činnost příkazem CALL C8ØØ v režimu MONITOR :

CC8ØØ

Podprogram na adrese C8Ø3H umožňuje vypsat na obrazovku text, na jehož první písmeno ukazuje adresa daná párovým registrem HL . Zkusme si ukázku :

Nastavme v režimu MONITOR příkazem CHANGE párový reristr HL na hodnotu 1ØØØH a proveďme následující substituce příkazem SUBST 1ØØØ :

```
S 1ØØØ : 41      ... ASCII kód A
  1ØØ1 : 48      ...    -"-    H
  1ØØ2 : 4F      ...    -"-    O
  1ØØ3 : CA      ...    -"-    J inverzně
  1ØØ4 : ØD      ...    -"-  } (CRLF kursoru)
  1ØØ5 : ØA      ...    -"-  }
  1ØØ6 : ØØ      ... koncová značka
```

Upozornění - poslední písmeno musí být zadáno inverzně ač se vytiskne neinverzně, nastavení kursoru na nový řádek (CRLF) není povinné, povinná je však koncová značka „ØØ".

Nyní spustíme program příkazem CALL CØØ3

CC8Ø3

a na obrazovce se vypíše

AHOJ

Zavoláme-li další odskokový program příkazem CALL CØØ6

CC8Ø6

vypíše se na obrazovku hlášení BREAK READY , čehož můžeme využít při odlaďování vlastních programů ve strojovém kódu, kdy se nám náš program přeruší a odskočí do BASICu.

Odskokový vektor počínaje adresou CØØ9H není využit.

Na adrese C85ØH se provede komparace s prvním bytem modulu kresliče (na adrese CØØØH)

Adresa (H):	Instrukce (H):	Mnemokód instrukce:
C84D :	3A ØØ CØ	LDA CØØØH
C85Ø :	FE 3E	CPI 3EH

Je-li zasunut kreslič, posune se oblast systémových proměnných BASICu z adresy 169H o další byty pro systémové proměnné kreslič až na adresu 21CH . (Zde bychom si dovolili odbočit a připomenout, že např. tzv. rychlonahrávky se zasunutým modulem kresli-

če a bez modulu kresliče se liší a program v nesprávné konfiguraci nelze do počítače nahrát a spustit (!))

Podprogramem na adrese C879H

C879 : CD 71 CD CALL CD71H

proběhne inicializace obdobně jako při příkazu BASICu „SCRATCH". Instrukcemi od adresy C882H

Adresa (H):	Instrukce (H):	Mnemokód instrukce:
C882 :	21 27 E5	LXI H,E527H
C885 :	C3 84 CC	JMP CC84H
⋮		⋮
CC84 :	CD 7D DB	CALL DB7DH

se vypíše na obrazovku „BASIC"

Obdobným podprogramem počínaje adresou CCCEH se na obrazovku ještě připíše „READY". A nyní se program (obdobně jako v režimu MONITOR) zastaví na vstupní rutině (počínajíc adresou CCD4H) vstupu znaků z klávesnice. Což je nám známý stav mikropočítače IQ 151, kdy nás blikající kurzor upozorňuje, že očekává od nás vstup požadované činnosti.

4.2.2.2. Vstupní buffer

Vložme z klávesnice IQ 151 následující posloupnost znaků :

ABCDEFGHIJKL

Tuto vstupní posloupnost znaků (případně i např. klíčová slova BASIC) si počítač zapsal na obrazovku, ale též si je zapsal do pracovních buněk, kterým se říká vstupní buffer (vstupní zásobník, zápisník, aj.). Zkusme ho objevit. Nechme na obrazovce naši posloupnost znaků A,B,C,..., ale neodesílejte ji (!). Přejděme do režimu MONITOR stiskem tlačítka BREAK a opět nám již dobře známým způsobem zobrazme na obrazovce první KB paměti RWM počínaje adresou ØØØØH příkazem MOVE :

MØØØØ,Ø3FF,ECØØ

V horní polovině obrazovky objevíme naši posloupnost znaků ABCDEFG.
Příkazem DISPLAY (případně SUBST) v režimu MONITOR zjistíme, že tento vstupní buffer (zásobník) začíná na adrese 5ØH a jeho velikost je 8Ø znaků. Zde si zpětně uvědomíme, že tedy náš programový řádek v režimu BASIC může mít maximálně 8Ø znaků včetně čísla řádku a mezer a Rovněž si uvědomme, že se v tomto vstupním bufferu provádí editace pomocí DELATE CHARACTER (DC) a INSERT CHARACTER (IC) .

Odešleme-li přece jenom naši posloupnost znaků ABCDEFG... tlačítkem CR, počítač nám oznámí "Ø1 ERROR", tj. "nerozumí" tomuto klíčovému slovu, nemá ho ve svém souboru klíčových slov. Zkusme nyní "objevit" tento soubor klíčových slov jazyka BASIC 6.

4.2.2.3. Soubor klíčových slov a jejich kódy

Proveďme zobrazení prvního KB programového vybavení BASIC 6, které začíná na adrese C8ØØH, příkazem MOVE v režimu MONITOR :

MC8ØØ,CBFF,ECØØ

Na obrazovce bezpečně objevíme nám známá klíčová slova jazyka BASIC. Soubor klíčových slov, či jakýsi slovník jazyka BASIC 6, začíná takto:

E N D F O R N E X T D A T A I N P U T D I M R E A D L E T G O T O

Všimněte si, že poslední písmeno je zapsáno inverzně, což bude ještě objasněno.

Objevíme zde i klíčová slova jako

V E C T , O R G , P O I N T , S I Z E , aj.,

což jsou klíčová slova pro řízení kreslíče (MINIGRAF Ø5Ø7, XY 413Ø).

Tabulku klíčových slov jsme tedy našli poměrně snadno. Zkusme teď ještě něco jiného. Pro jistotu stiskněme tlačítko RESET a vložme z klávesnice do počítače např. takovýto programový řádek

```
1Ø INPUT A
```


(Kdo chce, může se ve vstupním bufferu (zásobníku) přesvědčit, že je tam tento vstup zapsán takto znakově.) Tento programový řádek odešleme (CR) a podíváme se na vstupní buffer (zásobník) známým způsobem v režimu MONITOR:

MØØØØ,Ø3FF,ECØØ

Povšimněte si, že nám původní znakový zápis změnil na kódovaný. Navíc se nám tento programový řádek ještě uložil od adresy 16AH (v případě, že máme IQ 151 pouze s moduly VIDEO 32 a BASIC-6), resp. od adresy 21DH jestliže máme zasunut modul pro kreslič.
Ve vstupním bufferu (zásobníku) nyní máme tento obsah (ověřeno pomocí SUBST)

```
S 5Ø  :  84   ... interní kód příkazu INPUT
  51  :  41   ... ASCII kód „A"
  52  :  ØØ
  53  :  ØØ
  54  :  ØØ
```

A programový řádek se též uložil do paměti RWM od adresy 16AH známým /14/, /25/ způsobem (ověřte si pomocí SUBST)

```
S 16A : 71 }
  16B : Ø1 } = Ø171H ...odkaz na adresu, kde začíná
                        další řádek příkazu BASIC-6
  16C : ØA }
  16D : ØØ } = ØØØAH ...číslo řádku 1Ø(=AH)
  16E : 84   ... kód příkazu INPUT
  16F : 41   ... kód „A"
  17Ø : ØØ   ... koncová značka řádku
  171 : ØØ }
  172 : ØØ } ... koncová značka programu
```

V případě více řádků programu (např. 1Ø,2Ø,3Ø) se např. programový řádek č.15 samozřejmě zařadí mezi programové řádky 1Ø až 2Ø.

Zvláště si uvědomte, že program vlastně neexistuje v počítači ve znakovém tvaru, tak jak ho zadáme z klávesnice, ale že je

uložen v jisté kódované podobě. Např. příkaz INPUT není uložen jako posloupnost znaků I,N,P,U,T, ale jako kód 84H . Pozn.- zpět při listingu, či při zápisu na magnetofon je tomuto kódu 84h opět přiřazena posloupnost znaků I,N,P,U,T.

Zkusme zjistit kódy všech příkazů a funkcí v jazyku BASIC-6 u IQ 151. (Předem upozorňuji, že tyto kódy příkazů jsou např. pro PMD-85, ZX-81 aj. jiné (!). Není tedy možný přenos programů (jako byteové struktury) z počítače na počítač).

Napišme takovýto programový řádek:

```
1Ø END:FOR:NEXT:DATA:INPUT:DIM:READ:LET:GOTO:RUN:IF
```

A podívejme se jak se nám tento „program" uložil do paměti (příkazem SUBST v režimu MONITOR)

```
S16A : 7B }
 16B : Ø1 } odkaz na adresu, kde začíná další programový řádek
 16C : ØA }
 16D : ØØ } číslo řádku 1Ø          (1Ø=AH)
 16E : 8Ø ... kód END
 16F : 3A ... kód dvojtečky (:)
 17Ø : 81 ... kód FOR
 171 : 3A ... kód (:)
 172 : 82 ... kód NEXT
 173 : 3A ... kód (:)
 174 : 83 ... kód DATA
 175 : 3A ... kód (:)
 176 : 84 ... kód INPUT
 177 : ØØ ... koncová značka řádku
 178 : ØØ }... koncová značka programu
 179 : ØØ }
```

Všimněme si, že kódy jednotlivých klíčových slov jazyka BASIC 6 rostou od 8ØH postupně tak, jak jsou uspořádány za sebou klíčová slova jazyka BASIC 6 v tabulce. Kdo nevěří, může si to odzkoušet pro všechny příkazy a funkce jazyka BASIC 6.

4.2.2.4. Souvislost klíčových slov, kódů a adres podprogramů

Známe již klíčová slova, jejich kódy a nyní budeme hledat odskokovou tabulku a počátečními adresami pro jednotlivé příkazy a funkce BASIC 6. Víme již, např. z rozboru MONITORU, že bývá často umístěna v okolí tabulky slov jazyka. Tabulka slov BASIC 6 končí na adrese CA36H koncovou značkou ØØH. Poslední slovo je PTR ; ověřte si sami pomocí SUBST na adresách CA3ØH až CA34H.

Od adresy CA35H začíná "jakási" tabulka : ověřte pomocí SUBST

```
S CA35 : 82 }
  CA36 : DØ } ... DØ82H ... poč. adresa END
  CA37 : 8Ø }
  CA38 : CF } ... CF8ØH ... poč. adresa FOR
  CA39 : 55 }
  CA3A : D4 } ... D455H ... poč. adresa NEXT
  ....   .. }      atd.                  atd.
```

Tabulka odskokových adres příkazů jazyka BASIC a seznam slov jazyka BASIC bývají nejčastěji uspořádány tak, že prvnímu slovu (příkazu) v tabulce slov odpovídá první odskoková adresa tohoto příkazu (tak je tomu i u BASIC-6), případně též může odpovídat prvnímu slovu z tabulky poslední adresa odskoku (např. již známý MONITOR v IQ 151). Přiřazení jednotlivých odskokových adres k jednotlivým příkazům bude probíhat bez problému až do klíčového slova CLS . Poté následuje několik klíčových slov (např. TAB, TO, SPC, FN, THEN, STOP aj), které nemají vlastní procedury ve strojovém kódu, neboť jsou součástí celého příkazu - např. FOR - TO - - STEP se realizuje jako celek od odskokové adresy FOR , dále např. IF-THEN , DEF-FN , PRINT-TAB , PRINT-SPC , aj. Část odskokové tabulky pro funkce nalezneme též ještě (ověřte si pomocí DISPLAY či SUBST) před seznamem klíčových slov jazyka BASIC-6 počínaje adresou C888H . Souhrnně je vše včetně interních kódů příkazů a funkcí vyjádřeno v tab.4.1.

Klíčové slovo BASIC-6	Interní kód IQ 151	Počáteční adresa	Poznámka
END	80H	D082H	
FOR	81H	CF80H	
NEXT	82H	D455H	
DATA	83H	D1B2H	
INPUT	84H	D370H	
DIM	85H	D83AH	
READ	86H	D399H	
LET	87H	D1C9H	
GOTO	88H	D17EH	
RUN	89H	D162H	
IF	8AH	D23BH	
RESTORE	8BH	D035H	
GOSUB	8CH	D16EH	
RETURN	8DH	D19CH	
REM	8EH	D1B4H	
STOP	8FH	D080H	
OUT	90H	DD86H	
ON	91H	D21DH	
GET	92H	DD9BH	
POKE	93H	DE0BH	
PRINT	94H	D25CH	
LPRINT	95H	CB34H	
DEF	96H	D9C9H	
CONST	97H	D0ABH	
LIST	98H	CF31H	
LLIST	99H	CF29H	
CLEAR	9AH	D117H	
PTAPE	9BH	CAE1H	
MLOAD	9CH	CB00H	
PLIST	9DH	CAF0H	
MSAVE	9EH	CB24H	
SCRATCH	9FH	CD70H	

Tab.4.1. : Klíčová slova BASIC-6, jejich interní kódy a počáteční adresy jejich rutin

Klíčové slovo BASIC-6	Interní kód IQ 151	Počáteční adresa	Poznámka
CALL	A0H	E532H	
BYE	A1H	CAD3H	
WAIT(	A2H	E4DBH	
AUTO	A3H	CB62H	
MEM	A4H	D6EBH	
MOV	A5H	E60BH	
VECT	A6H	E614H	
POINT	A7H	E61DH	
ORG	A8H	E626H	
SPEED	A9H	E632H	
NARROW	AAH	E64CH	
WIDE	ABH	E64FH	
WRITE	ACH	E659H	
SIZE	ADH	E66BH	
FREE	AEH	D71BH	
UNPLOT	AFH	D655H	
PLOT	B0H	D657H	
CLS	B1H	D650H	
TAB(	B2H		součást PRINT
TO	B3H		součást FOR
SPC(	B4H		součást PRINT
FN	B5H		součást DEF
THEN	B6H		součást IF
NOT	B7H		
STEP	B8H		součást FOR
+	B9H	E1FEH	
-	BAH	DE29H	
*	BBH	DF68H	
/	BCH	DFC0H	
^	BDH	E311H	
AND	BEH	D794H	
OR	BFH	D793H	

Tab.4.1. - Pokračování

Klíčové slovo BASIC-6	Interní kód IQ 151	Počáteční adresa	Poznámka
>	CØH		
=	C1H		
<	C2H		
SGN	C3H	EØ7AH	
INT	C4H	E13EH	
ABS	C5H	EØ9ØH	
USR	C6H	DØE8H	
INP	C7H	DD7CH	
POS	C8H	D9B4H	
SQR	C9H	E3ØBH	
RND	CAH	E3EØH	
LOG	CBH	DF2AH	
EXP	CCH	E34FH	
COS	CDH	E413H	
SIN	CEH	E419H	
TAN	CFH	E47AH	
ATN	DØH	E48FH	
PEEK	D1H	DEØ4H	
LEN	D2H	DCE8H	
STRŠ	D3H	DAF3H	
VAL	D4H	DDE9H	
ASC	D5H	DCF7H	
CHRŠ	D6H	DDØ7H	
LEFTŠ	D7H	DD17H	
RIGHTŠ	D8H	DD47H	
MIDŠ	D9H	DD51H	
INKEYŠ	DAH	D624H	
HEX(	DBH	D5FDH	
PI	DCH	D645H	
BYTE(	DDH	D5DEH	
WORD(	DEH	D5ECH	
PTR(	DFH	D5D7H	

Tab.4.1. - Pokračování

Zkusme si ověřit, že jsme přiřadili správně odskokové adresy k jednotlivým příkazům. Jednoduchým způsobem to nelze ověřit u všech, nejsnadněji se to ověří u příkazů, které nevyžadují parametry. Jednotlivé příkazy se budeme pokoušet vyvolat pomocí příkazu CALL v režimu MONITOR . Např. LIST (viz. tab.4.1.) začíná na adrese CF31H . Napíšme si do počítače krátký program v jazyce BASIC, přejděte do režimu MONITOR a proveďte CALL CF31

CCF31 ..... (LIST)

a uvidíte, že se vám odstartuje listing programu.
Dále si obdobně ověřme příkaz BYE příkazem CALL CAD3

CCAD3 ..... (BYE),

či příkaz MEM (pomocí CALL DØEB)

CDØEB ..... (MEM),

případně příkaz CLS (pomocí CALL D65Ø)

CD65Ø ..... (CLS).

Zajímavé je též ověření, případně i využití, spuštění BASIC programu pomocí RUN . Napišme si krátký program v BASICu; nutné je, aby tento program začínal číslem řádku Ø (!)
Vyzkoušejme CALL D162

CD162 ..... (RUN) (první řádek musí mít číslo Ø)

a program se nám spustí.
Vyzkoušeli jsme několik příkazů a ověřili jsme správné přiřazení odskokových adres. Odzkoušení některých jiných příkazů nechávám na vyspělém uživateli IQ 151. Ostatní odskokové adresy jsou ověřeny z dokumentace /29/, či /28/.

Vrátíme se k poznatku, že poslední písmeno klíčových slov BASIC-6 v tabulce slov bylo inverzně (tj. jeho ASCII kód byl

o 8ØH větší, či ještě jinak řečeno jeho nejvyšší bit č.7 byla "1"). Toto např. u MONITORU nebylo potřeba, neboť všechny příkazy (slova) měla stejnou délku - jeden znak. BASIC-6 má však slova s různou délkou. Princip vyhledání klíčového slova a přiřazení počáteční adresy je též obdobný jako v našem MONITORU. Klíčové slovo BASICu-6 zapsané např. pomocí klávesnice do vstupního bufferu (zásobníku) se po odeslání tlačítkem (CR) začne pěkně popořádku srovnávat se seznamem slov v počítači IQ 151. Každý „přechod" přes jednotlivá klíčová slova „signalizuje" inversní znak na konci každého slova. Zároveň s postupným prohledáváním tabulky se postupně o jedničku zvyšuje interní kód příkazu či funkce (počínaje od 8ØH - kód END) a zároveň se posouváme v odskokové tabulce. V případě, že je nalezeno klíčové slovo z podmnožiny IQ 151, máme tedy jak počáteční adresu, tak i interní kód příkazu či funkce. V případě, že takové slovo počítač nezná, pak na nás „pípne" a oznámí nám, že takové slovo nezná - Ø1 ERROR (syntaktická chyba).

Trochu by se nám měla tímto ozřejmit interpretační činnost našeho jazyka. Počítač zjistí v programu klíčové slovo, odskočí si na zjištěnou adresu vykonat požadovanou činnost, dále se vrátí zpět do hlavního programu, nalezne další klíčové slovo a opět si odskočí „interpretovat" požadovanou činnost, atd.

Tabulku slov takto uloženou v IQ 151 využívá též procedura LIST (případně LLIST, či PLIST). Víme již, že BASICovský program je uložen v kódované formě, a že jednotlivým klíčovým slovům jazyka BASIC odpovídají jejich interní kódy. Při listingu jsou však opět tyto interní kódy rozepsány na znakovou formu (např. kód 8ØH ... END).

Povšimněte si, že některá klíčová slova v tab.4.1. jsou psána i se závorkou (např. PTR(, HEX(aj.). Dále je možno též upozornit, že některá slova pro kreslič mají společné klíčové slovo (např.

MOV je společné pro MOVA i MOVR) atd. Tabulku 4.1. nechť využije čtenář dle svých schopností.

4.2.2.5. Systémové proměnné

Na závěr o programovém vybavení BASIC-6 se ještě zmíníme o některých systémových proměnných BASIC-6. Tak jako MONITOR /14/ potřeboval některé pracovní buňky pro meziuložení výsledků či např. buňky pro určení stavu počítače (grafika, inverse aj.), potřebuje i BASIC-6 své pracovní systémové proměnné. Nemohou být umístěné libovolně, (aby se např. nepřekrývaly s uživatelskou oblastí pro programy BASIC) a musí to být paměť typu RWM , aby se informace nechala zapsat i přečíst.

Již při počátečním studiu BASIC-6 jsme objevili vstupní buffer (zásobník). Ten začíná na adrese 5ØH a končí na adrese 9FH , má tedy délku 6Ø bytů. Ve /14/ a /25/ jsou čtenáři upozorněni na systémové proměnné na adresách A4H až A7H . Na A4H a A5H je uložena počáteční adresa tzv. oblasti USR . Její koncová adresa je uložena na adresách A6H a A7H . Připomeňme, že počáteční adresa USR oblasti se posouvá buď přímou substitucí na adresách A4H a A5H , či BASIC příkazem CLEAR parametr pro oblast STRING , parametr pro oblast USR.

Asi nejzajímavější systémové proměnné BASIC-6 zkusíme „objevit" na následujícím BASIC programu.

```
 5 DIM E1(3,3):CLS
1Ø INPUT A1,B1,C1
2Ø PRINT A1,B1,C1
3Ø INPUT D1$
4Ø PRINT D1$
5Ø FOR I1=1 TO 3
6Ø FOR J1=1 TO 3
```

```
 7Ø READ E1(I1,J1)
 8Ø DATA 1,2,3,4,5,6,7,8,9,1Ø
 9Ø PRINT "E1("I1","J1")="E1(I1,J1)
1ØØ NEXT J1
11Ø NEXT I1
```

Program vložte do mikropočítače IQ 151 tak, jak je zapsán (i s mezerami !). Pozn.: jinak nebude souhlasit obsah na adresách DØH a D1H , kde je ukazatel konce BASIC programu, či přesněji řečeno ukazatel začátku jednoduchých proměnných, jak bude dále vysvětleno.

Po vložení programu, aniž bychom ho spustili, si prohlédneme obsah paměťových míst na adresách DØH až D7H - např. pomocí příkazu SUBST :

Adresa (H) :	Obsah (H) :	
DØ :	28	... Ø228H
D1 :	Ø2	
D2 :	28	... Ø228H
D3 :	Ø2	
D4 :	28	... Ø228H
D5 :	Ø2	
D6 :	69	... Ø169H
D7 :	Ø1	

Ukládání BASIC programu do paměti počítače bylo již popsáno v /14/. Z tzv. rychlonahrávek víme, že obsah adres DØH a D1H udává konec vloženého programu. Ve /14/ je rovněž vysvětleno ukládání jednoduchých proměnných a stringových proměnných do paměti počítače. Méně znalý čtenář nechť se raději k této problematice sám vrátí.

Spustíme náš vložený program (RUN) a vložíme požadované INPUTy např. takto :

```
: 1
: 2
: 3
: AHOJ
```

Přejdeme do režimu MONITOR a prohlédneme si opět obsah adresových míst DØH až D7H :

Adresa (H) :	Obsah (H) :	
DØ :	28	... Ø228H
D1 :	Ø2	
D2 :	4C	... Ø24CH
D3 :	Ø2	
D4 :	95	... Ø295H
D5 :	Ø2	
D6 :	F1	... Ø1F1H
D7 :	Ø1	

Vidíme, že se systémové proměnné na adresách DØH a D1H nezměnily, avšak systémové proměnné na adresách D2H až D7H se změnily. Předpokládáme, že uložení ukázkového programu do paměti počítače od adresy 16AH (opět připomínáme bez zasunutého modulu pro kresliče !) již uživatelé znají /14/. Zkusme tedy zjistit obsah paměťových míst počínaje adresou konce našeho BASIC programu . Dále se pokusme zjistit též obsahy paměťových míst počínaje adresou Ø24CH, 0295H, resp. obsah paměťového místa Ø1F1H nejlépe příkazem SUBST či DISPLAY. Po provedení příkazu SUBST Ø228 dostaneme následující obsazení paměťových míst :

Adresa (H):		Obsah (H):	
Ø228	:	31	... 1
Ø229	:	41	... A
Ø22A	:	ØØ	...1
Ø22B	:	ØØ	
Ø22C	:	ØØ	
Ø22D	:	81	
Ø22E	:	31	... 1
Ø22F	:	42	... B
Ø23Ø	:	ØØ	...2
Ø231	:	ØØ	
Ø232	:	ØØ	
Ø233	:	82	
Ø234	:	31	...1
Ø235	:	43	...C
Ø236	:	ØØ	...3
Ø237	:	ØØ	
Ø238	:	4Ø	
Ø239	:	82	
Ø23A	:	B1	...1
Ø23B	:	44	...D
Ø23C	:	Ø4	...délka „AHOJ"
Ø23D	:	FF	...oddělovač
Ø23E	:	9D	...adresa uložení AHOJ
Ø23F	:	7F	
Ø24Ø	:	31	...1
Ø241	:	49	...I
Ø242	:	ØØ	4 ...počet průchodů I1
Ø243	:	ØØ	
Ø244	:	ØØ	
Ø245	:	83	

Adresa (H):		Obsah (H):	
Ø246	:	31	. 1
Ø247	:	4A	...J
Ø248	:	ØØ	...4..počet průchodů J1
Ø249	:	ØØ	
Ø24A	:	ØØ	
Ø24B	:	83	
Ø24C	:	31	...1
Ø24D	:	45	...E
Ø24E	:	45	...počet bytů pole E1 dle deklarace DIM E1
Ø24F	:	ØØ	
Ø25Ø	:	Ø2	
Ø251	:	Ø4	
Ø252	:	ØØ	
Ø253	:	Ø4	
Ø254	:	ØØ	
Ø255	:	ØØ	
Ø256	:	ØØ	
Ø257	:	ØØ	
⋮			
Ø29Ø	:	83	
Ø291	:	ØØ	
Ø292	:	ØØ	
Ø293	:	1Ø	
Ø294	:	84	
Ø295	:	ØØ	
296	:	FF	

Od adresy Ø1F1H je toto obsazení:

Ø1F1	:	2C	... čárka (,)
Ø1F2	:	31	... 1 ...1Ø
Ø1F3	:	3Ø	... Ø
Ø1F4	:	ØØ	

Připomeňme si, že od adresy Ø228H (tj. konec BASIC programu) jsou postupně zapsané naše tři jednoduché proměnné A1=1, B1=2, C1=3 (zopakujte si ve /14/). Od adresy Ø23AH je uložena stringové proměnná D1$="AHOJ" - ve tvaru viz /14/ jeden byte délka stringu (= Ø4), jeden byte oddělovač (=FF) (pozn. může být i jiný) a dále dva byty určující adresu, kde je uložen obsah stringu. Přesvědčte se, že od adresy 7F9DH je uloženo postupně 41H, 4BH, 4FH, 4AH tj. A,H,O,J . Čtenář by si měl uvědomit souvislost s funkcí PTR jazyka BASIC.

Dále následuje od adresy Ø24ØH označení proměnné cyklu I1 a následující počet průchodů cyklem, totéž od adresy Ø246H platí pro proměnnou cyklu J1 . Ověřte si změnou hodnot v cyklu změnu uvedených bytů.

Na adrese Ø24CH začíná uložení pole konstant E1 . Na následujících dvou bytech (Ø24EH,Ø24FH) je uložen počet bytů pole E1, které zaberou jednak prvky pole a jednak jeho kódové označení. Počet bytů pole při deklaraci pole DIME1(3,3) je ØØ45H = 69 decimálně. Uvedená deklarace je pro pole (Ø až 3)x(Ø až 3) tj. 4x4. Uvážíme-li, že každé číslo je uloženo na 4 bytech potom vám vychází pro počet bytů :

$$69 = 4x4x4+5$$

Sami si ověřte jinou deklaraci pole.
Dále již nebudeme probírat jednotlivé byty deklarovaného pole, bude nám jistě stačit představa, že jsou zde uloženy hodnoty prvků pole E1.

Na závěr si ještě všimněme obsahu od adresy Ø1F1H . Nalezneme zde kód čárky, dále číslo 1Ø . Jestliže si uvědomíme, že adresa Ø1F1H je vlastně v našem BASIC programu, poznáme, že tato adresa ukazuje na další hodnotu zapsanou jako DATA. („1Ø" je hodnota,

která je úmyslně nadbytečná; eventuální další příkaz READ by načetl právě „1Ø" jako první).

Shrňme stručně tyto zjištěné poznatky:

- systémové proměnné na adresách DØH a D1H udávají začátek oblasti uložení jednoduchých proměnných (resp. konec BASIC programu)
- systémové proměnné na adresách D2H a D3H udávají začátek oblasti uložení polí (indexovaných proměnných)
- systémové proměnné na adresách D4H a D5H udávají konec oblasti polí resp. též vrch obsazené RWM
- systémové proměnné na adresách D6H a D7H udávají polohu posledních vybraných dat ze souboru DATA

Zajímavé praktické důsledky vyplynou z následujících příkladů. Pořiďme si následující tři nahrávky zkušebního BASIC programu, který jsme nechali proběhnout se stejnými vstupy INPUT : 1,2, 3,AHOJ, avšak nahrávky v režimu MONITOR (WRITE):

1. WDØ,228,CAD6 pozn. 228H je obsah adres DØH,D1H
2. WDØ,24C,CAD6 pozn. 24CH je obsah adres D2H,D3H
3. WDØ,295,CAD6 pozn. 295H je obsah adres D4H, D5H (CAD6H je skok do BASICu)

Vypněme na chvíli počítač. Programy postupně nahrajeme příkazem LOAD (L) a aniž bychom program spustili příkazem RUN (!) vyzkoušejme v přímém módu tyto tisky

PRINT A1,B1,C1,D1$,E1(2,2)

V nahrávce č.1 počítač IQ 151 nezná ani jednu z uvedených konstant.

V nahrávce č.2 počítač zná hodnoty A1,B1,C1,D1$ přiřadí 4 znaky „FF" (proč? - obsah stringu byl uložen daleko na konci v RWM). E1(2,2) přiřadí Ø.

V nahrávce č.3 počítač bude okamžitě znát totéž co ve 2.nahrávce

a navíc bude znát všechny prvky pole E1. Umíme tedy nahrát samotný BASIC program, případně BASIC program + jednoduché proměnné, případně BASIC program + indexované proměnné (pole).

Další zajímavá systémová proměnná je na adrese EAH. Podle jejího obsahu se rozliší, na které periferie se bude provádět např. PRINT či LIST případně INPUT. Z rozboru procedury LIST příp.LLIST, či PLIST či MSAVE dojdeme k poznatku, že při obsahu adresy EAH :

Adresa (H):		Obsah (H):	
EA	:	ØØ	se provede LIST na obrazovku
EA	:	Ø1	se provede LIST na děrnou pásku
EA	:	Ø2	se provede LIST na magnetofon
EA	:	Ø3	se provede LIST na tiskárnu

Procedury LLIST,PLIST,MSAVE mají jako základ proceduru LIST a rozlišení periferie se provede právě přiřazením dané hodnoty systémové proměnné na adrese EAH . Ověřme si to na příkladu. Příkaz LIST začíná na adrese C731H . V našem příkladu použijeme až adresu CF36H . Napišme libovolný BASIC program, připravte si tento příkazový řádek v přímém režimu

POKE HEX(EA),Ø2:CALL HEX(CB1C):CALL HEX(CF36),číslo program.řádku

dále přepněte magnetofon na záznam a teprve nyní proveďte odeslání příkazového řádku. Na magnetofon se nám zaznamená listing programu od zvoleného programového řádku.

Další praktický příklad je obdobou příkazu DATA SAVE, který IQ 151 nemá. Napišme tento program

```
1Ø X(1,1)=1:X(1,2)=2
2Ø X(2,1)=3:X(2,2)=4
3Ø POKE HEX( EA ),Ø2:CALL HEX(CB1C)    ...přepnutí na magnetofon
4Ø FOR I=1 TO 2
5Ø PRINT I "DATA"X(I,1)","X(I,2)       ...zápis čísla řádku,DATA,
                                          hodnoty X(1,1), čárky a
6Ø NEXT I                                 hodnoty X(I,2)
```

Tento program po spuštění (RUN) provádí „tisk" hodnot pole X(I,J) na magnetofonovou pásku ve formě BASIC programu s čísly řádků 1,2 a zápisem DATA:

```
1 DATA 1, 2
2 DATA 3, 4
```

Sami si ověřte zpětné načtení záznamu příkazem MLOAD.

Při ověřování činnosti funkce RND bychom „objevili" čtyři pracovní buňky počínaje adresou EBH . Po zapnutí počítače, či po RESET je v těchto buňkách stejný inicializační obsah. (Sami si ověřte). Jistě jste si již všimli, že první „náhodné" číslo je vždy Ø.5Ø438. Objasnění bude zřejmé uvědomíte-li si, že toto číslo se vytvoří jakýmsi algoritmem z čísla daného obsahem buněk EBH, ECH, EPH, EEH . Tohoto poznatku můžeme využít např. při generování dlouhé posloupnosti pseudonáhodných čísel, jestliže budeme nuceni např. přerušit generování pseudonáhodných čísel. Po přerušení běhu programu (např. příkazem STOP či BREAK) si poznamenáme obsah systémových proměnných na adresách EBH až EEH . Před opětovným spuštěním (RUN) provedeme substituci na adresách EBH až EEH (příkazem SUBST v režimu MONITOR), kterou jsme si dříve poznamenali. Obsahu adres EBH až EEH se říké vsádka či též násada pseudonáhodného čísla.

Zvídavý uživatel IQ 151 by nalezl ještě mnohé jiné systémové proměnné BASIC-6, avšak jejich úplný výčet není v možnostech této publikace. Ještě se stručně zmíníme o některých systémových proměnných kresliče XY413Ø. Na adresách

16ØH, 161H	je adresa ukazující na záznam stringu (konkrétně adresa ukazuje na délku stringu)	
162H, 163H	je XX	z příkazu SIZE XX,XY,YX,YY
164H, 165H	je XY	
166H, 167H	je YX	
168H, 169H	je YY	
16AH, 16BH	je X	z příkazu ORG X,Y
16CH, 16DH	je Y	
16EH, 16FH	je PX	k určení okamžité polohy v souřadnicovém systému X,Y
17ØH, 171H	je PY	
172H, 173H	je rozlišení módu NARROW, WIDE	

Majitelé tohoto kresliče nechť si sami ověří uvedené systémové proměnné. (Pozn. pro MINIGRAF Ø5Ø7 je obsazení systémových proměnných částečně odlišné.)

V této kapitole jsme si měli možnost vyzkoušet mnohé poznatky o programovém vybavení BASIC-6. Obecné shrnutí je provedeno v následující kapitole.

4.3. Shrnutí základních poznatků o mikropočítači IQ 151

Na rozdíl od předcházejících kapitol 4.1. a 4.2. by souhrn dále uvedených poznatků o mikropočítači IQ 151 měl znát každý

uživatel, který to myslí s IQ 151 jen trochu vážně.

Všeobecná charakteristika mikropočítače IQ 151

Mikropočítač IQ 151 je modulární otevřený 8-bitový mikropočítač. Je vybudovaný na mikroprocesoru MHB 8Ø8Ø (světový ekvivalent má typové označení 8Ø8Ø). Základní vstupně výstupní periferie, se kterými mikropočítač umožňuje komunikaci, jsou běžný TV přijímač či TV monitor - užívá se jako alfanumerický displej, magnetofon (nejlépe kazetový) - užívá se jako vnější paměť mikropočítače a též se někdy mezi základní vstupní periferie zahrnuje i klávesnice, která je u IQ 151 membránová. Umožňuje základní dialog mezi uživatelem a mikropočítačem. Klávesnice umožňuje tisk celého souboru velkých a malých písmen, číslic aj. IQ 151 je v základní sestavě (s modulem VIDEO 32 a BASIC-6) vybaven pouze semigrafikou a grafickým zobrazením 64x64 bodů. Zobrazení může být bílé na černém pozadí, případně obráceně tzv. inversní zobrazení. Klávesnice umožňuje opakovaný tisk při stisknutém tlačítku (autorepert). Mikropočítač je vybaven jednoduchou zvukovou signalizací.

Technické řešení mikropočítače IQ 151

Technické řešení je založeno na mikroprocesoru MHB 8Ø8Ø. Mikropočítač je v základní sestavě vybaven 32 KB paměti RWM (tzv. dynamická paměť) a 8KB paměti ROM (EPROM) na jazyk BASIC-6 a 4KB paměti ROM (EPROM) na základní programové vybavení MONITOR. IQ 151 je vybaven technickými prostředky, které umožňují komunikaci se TV přijímačem, magnetofonem a klávesnicí. Rozšiřující sběrnice IQ 151 pro další moduly obsahuje mimo mikroprocesorových sběrnic (adresová, datová a řídící) též např. i napájení a některé jiné další výhodné signály. Mikropočítač IQ 151 je vybaven přerušovacím systémem. O tom, že IQ 151 je modulární systém, jsme se již zmínili. Minimální konfigurace, ve které je systém schopný provozu, je IQ151 + modul VIDEO 32 (resp . VIDEO 64). Uživatel i začátečník by měl mít představu o rozdělení adresového prostoru IQ 151. Měl by znát umístění MONITORU (F8ØØH až FFFFH), BASIC-6 (C8ØØ až E7FFH), VIDEO RAM (ECØØH až EFFFH) pro VIDEO 32, uživatelské RWM - zde by si uživatel měl uvědomit, že MONITOR i BASIC-6 aj. mají některé systémové proměnné, které jsou uloženy na počátku (od adresy ØØØØH do Ø169H-např. pro BASIC-6), případně na konci 32KB (uložení stringových proměnných, případně tzv. USR oblast). Pochopení významu jednotlivých signálů rozšiřující sběrnice IQ 151 doporučujeme těm uživatelům, kteří chtějí IQ 151 využít neštandardně k připojení některých dalších periferií.

Programové vybavení mikropočítače IQ 151

Mikropočítač IQ 151 je vybaven relativně komfortním základním programovým vybavením MONITOR, které má deset příkazů (S,D, F,M,L,W,G,C,X a R). Jejich význam by uživatel měl znát již z /14/. Vyšší programovací jazyk, který se prozatím užívá v IQ 151, je BASIC-6, což je varianta jazyka BASIC. Příkazy a funkce BASIC-6 nebudeme připomínat, uživatel se s nimi již seznámil v /23/, /25/. Mikropočítač dovede v režimu BASIC pracovat jak v přímém módu (tj. vykoná všechny příkazy (samozřejmě oddělené dvojtečkou) přímo po odeslání (CR); příkazový řádek nezačíná číslicí), tak v programovacím módu (tj. příkazový řádek (či zde programový řádek) začíná číslem řádku). Po odeslání (CR) programového řádku se naprogramovaná činnost neprovede, pouze se tento programový řádek zařadí na odpovídající místo v programové paměti.

O činnosti MONITORU by si měl uživatel odnést představu, že je to soubor podprogramů napsaných ve strojovém kódu. Programové vybavení MONITORU dovede poznat podle tabulky „slov" MONITORU,

je-li požadovaný příkaz ze souboru MONITOR. Je-li takový příkaz nalezen, skočí MONITOR na počáteční adresu požadovaného příkazu. Počáteční tzv. odskokovou adresu určí při prohledávání tabulky příkazů tak, že se zároveň při posouvání v tabulce „slov" (příkazů) zároveň posouvá v tabulce odskokových adres. Nalezne-li požadovaný příkaz, pak přenese do tohoto podprogramu parametry zadané z klávesnice a vykoná určenou činnost. Poté se opět vrátí do režimu MONITOR na podprogram vstup z klávesnice, kdy od uživatele očekává další příkazy. Mezi důležité podprogramy MONITORU patří programová obsluha vstupu z klávesnice a výstupu na obrazovku, ale též i inicializace mikropočítače IQ 151. Při inicializaci se počítač nastaví do přesně definovaného stavu, naprogramuje všechny programovatelné obvody IQ 151, případně i periferní obvody, zjistí si, které moduly má zasunuté, případně provede některé další nezbytné činnosti. Činnost samotného MONITORU (ať po zapnutí počítače, či při RESET, či po vykonání uživatelem požadované činnosti) se vždy zastaví na vstupním podprogramu čtení klávesnice, neboť pouze z klávesnice může začít další požadovaná činnost MONITORU.

Jestliže byl v IQ 151 zasunut i modul BASIC-6 přenesla se počáteční inicializace i do programového vybavení BASIC. Zde se obdobně inicializovaly další systémové proměnné BASIC a inicialisační činnost se zastavila až na vstupním podprogramu čtení klávesnice (v BASIC programovém vybavení). V BASIC programovém vybavení se znaky z klávesnice nejdříve vkládají do tzv. vstupního bufferu (zásobníku). Teprve po odeslání počítač zjistí začíná-li tato znaková posloupnost číslicí. Jestliže ano, pak ji pochopí jako programový řádek a provede její zakódování a zařazení do uživatelského programu BASIC. A jestliže nezačíná číslicí, provede požadovanou činnost ihned po odeslání v tzv. přímém režimu.

Uživatel, i začátečník, by měl znát, že program není uložen ve znakové formě tak, jak ho zadáváme z klávesnice, ale že je uložen v kódovaném tvaru.
Interpretace požadovaného BASIC příkazu či funkce se provede na obdobném principu podle tabulky „slov" BASIC a tabulky odskokových adres BASIC. Zadaný BASIC program řeší IQ 151 tak, že postupné (podle čísel řádků) prochází tento program a každý BASIC příkaz „interpretuje" podprogrem ve strojovém kódu, který je umístěn v ROM (EPROM).

Z činnosti programového vybavení MONITOR a BASIC jsme poznali nutnost některých systémových proměnných. Následující výčet považujeme za základní a měl by ho umět (alespoň v některých speciálních případech) využít i začátečník.

MONITOR:

ØØØ8H ... časová informace (2Ø ms)
ØØØ9H ... časová informace (256*2Ø ms)
ØØØAH ... časová informace (256 * 256 * 2Ø ms)
ØØØBH ... kód znaku, na nějž ukazuje kurzor
ØØØEH ... poloha kurzoru na řádce
ØØØFH ... číslo řádku na obrazovce, kde se nachází kurzor
ØØ1ØH ... grafický mód
ØØ11H ... inversní mód
ØØ13H ... délka stránky na obrazovce
ØØ14H ... počet odřádkování po (CR)
ØØ17H ... výška tónu zvukového generátoru
ØØ18H ... délka tónu zvukového generátoru
ØØ1CH ... délka meziblokové mezery při nahrávání na magnetofon

BASIC-6:

ØØA4H, ØØA5H ... horní adresa chráněné USR oblasti

ØØA6H, ØØA7H ... dolní adresa chráněné USR oblasti

ØØCEH, ØØCFH ... adresa odkud se ukládá BASIC program

ØØDØH, ØØD1H ... adresa konce BASIC programu (bez proměnných); adresa, kde začínají jednoduché proměnné

ØØD2H, ØØD3H ... adresa začátku oblasti indexovaných proměnných

ØØD4H, ØØD5H ... adresa konce oblasti indexovaných proměnných; konec obsazené RWM paměti

ØØEAH ... modifikace vstupní, výstupní periferie

V této kapitole jsme chtěli shrnout nejzákladnější poznatky z konstrukce a programového vybavení IQ 151, které by měl každý uživatel bezpečně ovládat. Znalost programování ve vyšším jazyce BASIC a zvládnutí příkazů MONITORU se již považuje za samozřejmost.

5. Závěr

Probrali jsme si tedy základy konstrukce a činnosti mikropočítače, zvláště pak školního mikropočítače IQ 151. Začátečníci budou zřejmě zahrnuti množstvím informací, naopak vyspělí uživatelé zde naleznou zřejmě jenom málo pro ně nových informací. Publikace je určena pro středně pokročilé uživatele mikropočítače a měla by poskytnout populárně odbornou odpověď na některé otázky z činnosti mikropočítače.

Uživatel, který porozuměl celé problematice z této publikace, ověřil si ukázkové a některé své příklady např. na mikropočítači IQ 151, sám pozná, že bude mezi "mikropočítačovými kolegy" považován za "znalce" např. mikropočítače IQ 151.

Publikace nechce a ani nemůže být kompletní servisní technickou a programovou dokumentací. Úpravy, opravy a vývoj mikropočítačů však přenechme raději profesionálním výrobcům. Na druhé straně však je možné i s uvedenými středně odbornými znalostmi např. připojit k mikropočítači mnohé jednoduché nestandardní periferie, či též kvalitativně lépe využít programové vybavení vašeho mikropočítače.

V závěru bychom chtěli opět zdůraznit, že publikace má velmi široký záběr od historie počítače až po technické a programové vybavení mikropočítače. Uživatelé nechť si berou z této publikace jenom ty informace, na které se sami "cítí".

A úplně na závěr bychom chtěli vyzvat všechny uživatele např. mikropočítače IQ 151, aby své zajímavé, vtipné a výhodné poznatky z technického nebo programového vybavení mikropočítače "nestrkali do šuplíku", ale aby své poznatky poskytli prostřednictvím článků a publikací i ostatním uživatelům mikropočítače.

Mnoho úspěchů a příjemných chvil s mikropočítačem vám přeje

autor

6. Literatura :

/1/ Havlík,R.: MSX - nový mikropočítačový standard, AR3,4/86, 103-104, 143-144

/2/ MSX - nový standard pro mikropočítače (zn.-hák), ST 8/84,294

/3/ Budínský,J.: Amatérské a osobní mikropočítače, ARA 2 až 8/80

/4/ Pecinovský,R.: Z historie výpočetní techniky, T 84-86

/5/ ST 2/86 zn.-hák-: Nejúspěšnější mikropočítače roku, (1986)

/6/ Deset let mikropočítačů, ARA 10/85, 382-383 převzato z Deutscher Dokumentartag 1983, Mnichov, (1984), 457-470

/7/ Mikropočítače - jak to začalo, ARA 11/85, 424

/8/ Oswald,L.: Der Microcomputer im Kleinbetrieb, Holzkirchen, NSR, (1981)

/9/ rk : Die Deutschen auf dem Computer-Trip, CHIP 10, 50-52, (1984)

/10/ Hyan,J.,T.: Provozní systém CP/M, ARA 9/84, 340-342, (1984)

/11/ Klub vyvolených, T3, 50, zn.SŠ, (1986)

/12/ Sobotka,Z.: Otázky a odpovede z mikroprocesorov a mikropočítačov, ALFA, (1981)

/13/ Gvozdjak,L. a kol.: Počítače a programovanie, ALFA, (1985)

/14/ Feil,M.: Monitor IQ 151, Komenium, (1986)

/15/ Feil,M.: Strojový kód IQ 151, Komenium, (1986)

/16/ Zdeněk,J.: Technika mikropočítačů, ČSVTS, (1983)

/17/ Blatný,J. a kol.: Číslicové počítače, SNTL, (1982)

/18/ Dědina,B., Valášek,P.: Mikroprocesory a mikropočítače, SNTL, (1983)

/19/ Starý J.: Mikropočítač a jeho programování, SNTL, (1984)

/20/ Sobotka,Z., Starý,J.: Mikropočítače, ČSVTS, (1983)

/21/ Mikropočítače a mikroprocesory, AR 1-9/82

/22/ Smutný,E.: ARB 1 a 2/83

/23/ Jedlička,Z., Feil,M.: Basic pro začátečníky, Komenium,(1985)

/24/ Návod k obsluze a údržbě počítače IQ 151, ZPA Nový Bor,(1984)

/25/ Kollert,E.: Programování počítače IQ 151 v jazyku BASIC, Komenium, (1984)

/26/ Valášek,P.: Mikroprocesor 8080 a základní obvody, ČSVTS, (1982)

/27/ Vít,V. a kol.: Televizní technika, SNTL, (1979)

/28/ Mašek,L.: Zdrojový program **MONI**TOR IQ 151, ZPA Nový Bor (1985)

/29/ Mašek,L.: Zdrojový program BASIC-6 pro IQ 151, ZPA Nový Bor (1986)

/30/ Bayer,J., Bílek,J.: Mikroprocesory-úvod, skriptum ČVUT, (1983), 157 s

/31/ Kollert,E.: Výpočetní technika, SNTL, (1985), 171 s

/32/ Slípka,J.: Navrhování mikroprocesorových systémů, SNTL, (1985), 303 s

/33/ Budínský,J.: Polovodičové paměti a jejich použití, SNTL, (1977), 511 s

/34/ Dlabola,F.; Starý,J.: Syotémy s mikroprocesory a přenos dat, NADAS, (1984), 431 s

/35/ Kulič,V.: Člověk-učení-automat, SPN, (1984), 249 s

/36/ Borský,V., Podivín,L.: Technika počítačů 1, SNTL, (1982), 235 s

/37/ Jinoch,J., Müller,K., Vogel,J.: Programování v jazyku PASCAL, SNTL, (1985), 264 s

/38/ Machačka,I., Pavlů,J.: Programování v jazyku BASIC, SNTL, (1985), 212 s

/39/ Krištoufek,K. a kol.: Výpočetní a řídící technika, SNTL, (1986), 376 s

7. Dodatky

1. Tabulka 6.1. - Kód IQ 151 (částečně shodný s kódem ASCII)
2. Tabulka 6.2. - Instrukční kód mikroprocesoru 8Ø8Ø (uspořádaný abecedně)
3. Tabulka 6.3. - Instrukční kód mikroprocesoru 8Ø8Ø (uspořádaný vzestupně podle hexadecimální hodnoty)

Vysvětlivky:

a ... adresa (16 bitů)

d ... konstanta (8 bitů)

w ... konstanta (16 bitů)

* ... nastavuje příznaky CY, Z, S, P, AC

\+ ... nastavuje pouze CY

! ... nastavuje Z, S, P, AC

* ... pouze pro mikroprocesor 8Ø85

Pozn. : V tab. 6.2. a 6.3. jsou uvedeny kompletně všechny možné instrukce mikroprocesoru 8Ø8Ø. Někdy se pod termínem základní instrukce mikroprocesoru nerozumí např. konkrétní MOV A,B aj., ale instrukce MOV, představující celou skupinu přesunů, takže je základních instrukcí méně.

KÓD IQ 151 (částečně shodný s kódem ASCII)

norm. D	norm. H	zn.	inverse H	inverse D
0	00	●	80	128
1	01	○	81	129
2	02	→	82	130
3	03	←	83	131
4	04	↑	84	132
5	05	↓	85	133
6	06	◆	86	134
7	07	▶	87	135
8	08	◆	88	136
9	09	▒	89	137
10	0A	▒	8A	138
11	0B	▒	8B	139
12	0C	▒	8C	140
13	0D	▒	8D	141
14	0E	▒	8E	142
15	0F	▒	8F	143
16	10	▒	90	144
17	11	▒	91	145
18	12	▒	92	146
19	13	▒	93	147
20	14	▒	94	148
21	15	▒	95	149
22	16	▒	96	150
23	17	▒	97	151
24	18	▒	98	152
25	19	▒	99	153
26	1A	▒	9A	154
27	1B	▒	9B	155
28	1C	▒	9C	156
29	1D	▒	9D	157
30	1E	▒	9E	158
31	1F	▒	9F	159
32	20	(sp)	A0	160
33	21	!	A1	161
34	22	"	A2	162
35	23	#	A3	163
36	24	$	A4	164
37	25	%	A5	165
38	26	&	A6	166
39	27	'	A7	167
40	28	(	A8	168
41	29	)	A9	169
42	2A	*	AA	170
43	2B	+	AB	171
44	2C	,	AC	172
45	2D	-	AD	173
46	2E	.	AE	174
47	2F	/	AF	175
48	30	0	B0	176
49	31	1	B1	177
50	32	2	B2	178
51	33	3	B3	179
52	34	4	B4	180
53	35	5	B5	181
54	36	6	B6	182
55	37	7	B7	183
56	38	8	B8	184
57	39	9	B9	185
58	3A	:	BA	186
59	3B	;	BB	187
60	3C	<	BC	188
61	3D	=	BD	189
62	3E	>	BE	190
63	3F	?	BF	191
64	40	@	C0	192
65	41	A	C1	193
66	42	B	C2	194
67	43	C	C3	195
68	44	D	C4	196
69	45	E	C5	197
70	46	F	C6	198
71	47	G	C7	199
72	48	H	C8	200
73	49	I	C9	201
74	4A	J	CA	202
75	4B	K	CB	203
76	4C	L	CC	204
77	4D	M	CD	205
78	4E	N	CE	206
79	4F	O	CF	207
80	50	P	D0	208
81	51	Q	D1	209
82	52	R	D2	210
83	53	S	D3	211
84	54	T	D4	212
85	55	U	D5	213
86	56	V	D6	214
87	57	W	D7	215
88	58	X	D8	216
89	59	Y	D9	217
90	5A	Z	DA	218
91	5B	[	DB	219
92	5C	\	DC	220
93	5D	]	DD	221
94	5E	^	DE	222
95	5F	_	DF	223
96	60	`	E0	224
97	61	a	E1	225
98	62	b	E2	226
99	63	c	E3	227
100	64	d	E4	228
101	65	e	E5	229
102	66	f	E6	230
103	67	g	E7	231
104	68	h	E8	232
105	69	i	E9	233
106	6A	j	EA	234
107	6B	k	EB	235
108	6C	l	EC	236
109	6D	m	ED	237
110	6E	n	EE	238
111	6F	o	EF	239
112	70	p	F0	240
113	71	q	F1	241
114	72	r	F2	242
115	73	s	F3	243
116	74	t	F4	244
117	75	u	F5	245
118	76	v	F6	246
119	77	w	F7	247
120	78	x	F8	248
121	79	y	F9	249
122	7A	z	FA	250
123	7B	{	FB	251
124	7C	|	FC	252
125	7D	}	FD	253
126	7E	~	FE	254
127	7F	■	FF	255

Tab. 6.1.

Tab. 6.2. : Instrukční kód mikroprocesoru 8080 (uspořádaný abecedně)

symb.	kód	poz.	symb.	kód	poz.	symb.	kód	poz.	symb.	kód	poz.	symb.	kód	poz.	symb.	kód	poz.
ACI d	CE	"	CNZ a	C4		JMP a	C3		MOV D,C	51		MVI E,d	1E		RST 4	E7	
ADC A	8F	"	CP a	F4		JNC a	D2		MOV D,D	52		MVI H,d	26		RST 5	EF	
ADC B	88	"	CPE a	EC		JNZ a	C2		MOV D,E	53		MVI L,d	2E		RST 6	F7	
ADC C	89	"	CPI d	FE	"	JP a	F2		MOV D,H	54		MVI M,d	36		RST 7	FF	
ADC D	8A	"	CPO a	E4		JPE a	EA		MOV D,L	55		NOP	00		RZ	C8	
ADC E	8B	"	CZ a	CC		JPO a	E2		MOV D,M	56		ORA A	B7	"	SBB A	9F	"
ADC H	8C	"	DAA	27	"	JZ a	CA		MOV E,A	5F		ORA B	B0	"	SBB B	98	"
ADC L	8D	"	DAD B	09	+	LDA a	3A		MOV E,B	58		ORA C	B1	"	SBB C	99	"
ADC M	8E	"	DAD D	19	+	LDAX B	0A		MOV E,C	59		ORA D	B2	"	SBB D	9A	"
ADD A	87	"	DAD H	29	+	LDAX D	1A		MOV E,D	5A		ORA E	B3	"	SBB E	9B	"
ADD B	80	"	DAD SP	39	+	LHLD a	2A		MOV E,E	5B		ORA H	B4	"	SBB H	9C	"
ADD C	81	"	DCR A	3D	!	LXI B,w	01		MOV E,H	5C		ORA L	B5	"	SBB L	9D	"
ADD D	82	"	DCR B	05	!	LXI D,w	11		MOV E,L	5D		ORA M	B6	"	SBB M	9E	"
ADD E	83	"	DCR C	0D	!	LXI H,w	21		MOV E,M	5E		ORI d	F6	"	SBI d	DE	"
ADD H	84	"	DCR D	15	!	LXI SP,w	31		MOV H,A	67		OUT d	D3		SHLD a	22	
ADD L	85	"	DCR E	1D	!	MOV A,A	7F		MOV H,B	60		PCHL	E9		SIM *	30	
ADD M	86	"	DCR H	25	!	MOV A,B	78		MOV H,C	61		POP B	C1		SPHL	F9	
ADI d	C6	"	DCR L	2D	!	MOV A,C	79		MOV H,D	62		POP D	D1		STA a	32	
ANA A	A7	"	DCR M	35	!	MOV A,D	7A		MOV H,E	63		POP H	E1		STAX B	02	
ANA B	A0	"	DCX B	0B		MOV A,E	7B		MOV H,H	64		POP PSW	F1	"	STAX D	12	
ANA C	A1	"	DCX D	1B		MOV A,H	7C		MOV H,L	65		PUSH B	C5		STC	37	+
ANA D	A2	"	DCX H	2B		MOV A,L	7D		MOV H,M	66		PUSH D	D5		SUB A	97	"
ANA E	A3	"	DCX SP	3B		MOV A,M	7E		MOV L,A	6F		PUSH H	E5		SUB B	90	"
ANA H	A4	"	DI	F3		MOV B,A	47		MOV L,B	68		PUSH PSW	F5		SUB C	91	"
ANA L	A5	"	EI	FB		MOV B,B	40		MOV L,C	69		RAL	17	+	SUB D	92	"
ANA M	A6	"	HLT	76		MOV B,C	41		MOV L,D	6A		RAR	1F	+	SUB E	93	"
ANI d	E6	"	IN d	DB		MOV B,D	42		MOV L,E	6B		RC	D8		SUB H	94	"
CALL a	CD		INR A	3C	!	MOV B,E	43		MOV L,H	6C		RET	C9		SUB L	95	"
CC a	DC		INR B	04	!	MOV B,H	44		MOV L,L	6D		RIM *	20		SUB M	96	"
CM a	FC		INR C	0C	!	MOV B,L	45		MOV L,M	6E		RLC	07	+	SUI d	D6	"
CMA	2F		INR D	14	!	MOV B,M	46		MOV M,A	77		RM	F8		XCHG	EB	
CMC	3F	+	INR E	1C	!	MOV C,A	4F		MOV M,B	70		RNC	D0		XRA A	AF	"
CMP A	BF	"	INR H	24	!	MOV C,B	48		MOV M,C	71		RNZ	C0		XRA B	A8	"
CMP B	B8	"	INR L	2C	!	MOV C,C	49		MOV M,D	72		RP	F0		XRA C	A9	"
CMP C	B9	"	INR M	34	!	MOV C,D	4A		MOV M,E	73		RPE	E8		XRA D	AA	"
CMP D	BA	"	INX B	03		MOV C,E	4B		MOV M,H	74		RPO	E0		XRA E	AB	"
CMP E	BB	"	INX D	13		MOV C,H	4C		MOV M,L	75		RRC	0F	+	XRA H	AC	"
CMP H	BC	"	INX H	23		MOV C,L	4D		MVI A,d	3E		RST 0	C7		XRA L	AD	"
CMP L	BD	"	INX SP	33		MOV C,M	4E		MVI B,d	06		RST 1	CF		XRA M	AE	"
CMP M	BE	"	JC a	DA		MOV D,A	57		MVI C,d	0E		RST 2	D7		XRI d	EE	"
CNC a	D4		JM a	FA		MOV D,B	50		MVI D,d	16		RST 3	DF		XTHL	E3	

Tab.6.3. : Instrukční kód mikroprocesoru 8080 (podle hexadecimální hodnoty)

kód	symb.	dél.	kód	symb.	dél.	kód	symb.	dél.	kód	symb.	dél.	kód	symb.	dél.	kód	symb.	dél.	kód	symb.	dél.
00	NOP	1	2B	DCX H	1	56	MOV D,M	1	80	ADD B	1	AB	XRA E	1	D6	SUI d	2			
01	LXI B,w	3	2C	INR L	1	57	MOV D,A	1	81	ADD C	1	AC	XRA H	1	D7	RST 2	1			
02	STAX B	1	2D	DCR L	1	58	MOV E,B	1	82	ADD D	1	AD	XRA L	1	D8	RC	1			
03	INX B	1	2E	MVI L,d	2	59	MOV E,C	1	83	ADD E	1	AE	XRA M	1	D9	---	-			
04	INR B	1	2F	CMA	1	5A	MOV E,D	1	84	ADD H	1	AF	XRA A	1	DA	JC a	3			
05	DCR B	1	30	SIM *	1	5B	MOV E,E	1	85	ADD L	1	B0	ORA B	1	DB	IN d	2			
06	MVI B,d	2	31	LXI SP,w	3	5C	MOV E,H	1	86	ADD M	1	B1	ORA C	1	DC	CC a	3			
07	RLC	1	32	STA a	3	5D	MOV E,L	1	87	ADD A	1	B2	ORA D	1	DD	---	-			
08	---	-	33	INX SP	1	5E	MOV E,M	1	88	ADC B	1	B3	ORA E	1	DE	SBI d	2			
09	DAD B	1	34	INR M	1	5F	MOV E,A	1	89	ADC C	1	B4	ORA H	1	DF	RST 3	1			
0A	LDAX B	1	35	DCR M	1	60	MOV H,B	1	8A	ADC D	1	B5	ORA L	1	E0	RPO	1			
0B	DCX B	1	36	MVI M,d	2	61	MOV H,C	1	8B	ADC B	1	B6	ORA M	1	E1	POP H	1			
0C	INR C	1	37	STC	1	62	MOV H,D	1	8C	ADC H	1	B7	ORA A	1	E2	JPO a	3			
0D	DCR C	1	38	---	-	63	MOV H,E	1	8D	ADC L	1	B8	CMP B	1	E3	XTHL	1			
0E	MVI C,d	2	39	DAD SP	1	64	MOV H,H	1	8E	ADC M	1	B9	CMP C	1	E4	CPO a	3			
0F	RRC	1	3A	LDA a	3	65	MOV H,L	1	8F	ADC A	1	BA	CMP D	1	E5	PUSH H	1			
10	---	-	3B	DCX SP	1	66	MOV H,M	1	90	SUB B	1	BB	CMP E	1	E6	ANI d	2			
11	LXI D,w	3	3C	INR A	1	67	MOV H,A	1	91	SUB C	1	BC	CMP H	1	E7	RST 4	1			
12	STAX D	1	3D	DCR A	1	68	MOV L,B	1	92	SUB D	1	BD	CMP L	1	E8	RPE	1			
13	INX D	1	3E	MVI A,d	2	69	MOV L,C	1	93	SUB E	1	BE	CMP M	1	E9	PCHL	1			
14	INR D	1	3F	CMC	1	6A	MOV L,D	1	94	SUB H	1	BF	CMP A	1	EA	JPE a	3			
15	DCR D	1	40	MOV B,B	1	6B	MOV L,E	1	95	SUB L	1	C0	RNZ	1	EB	XCHG	1			
16	MVI D,d	2	41	MOV B,C	1	6C	MOV L,H	1	96	SUB M	1	C1	POP B	1	EC	CPE a	3			
17	RAL	1	42	MOV B,D	1	6D	MOV L,L	1	97	SUB A	1	C2	JNZ a	3	ED	---	-			
18	---	-	43	MOV B,E	1	6E	MOV L,M	1	98	SBB B	1	C3	JMP a	3	EE	XRI d	2			
19	DAD D	1	44	MOV B,H	1	6F	MOV L,A	1	99	SBB C	1	C4	CNZ a	3	EF	RST 5	1			
1A	LDAX D	1	45	MOV B,L	1	70	MOV M,B	1	9A	SBB D	1	C5	PUSH B	1	F0	RP	1			
1B	DCX D	1	46	MOV B,M	1	71	MOV M,C	1	9B	SBB E	1	C6	ADI d	2	F1	POP PSW	1			
1C	INR E	1	47	MOV B,A	1	72	MOV M,D	1	9C	SBB H	1	C7	RST 0	1	F2	JP a	3			
1D	DCR E	1	48	MOV C,B	1	73	MOV M,E	1	9D	SBB L	1	C8	RZ	1	F3	DI	1			
1E	MVI E,d	2	49	MOV C,C	1	74	MOV M,H	1	9E	SBB M	1	C9	RET	1	F4	CP a	3			
1F	RAR	1	4A	MOV C,D	1	75	MOV M,L	1	9F	SBB A	1	CA	JZ a	3	F5	PUSH PSW	1			
20	RIM *	1	4B	MOV C,E	1	76	HLT	1	A0	ANA B	1	CB			F6	ORI d	2			
21	LXI H,w	3	4C	MOV C,H	1	77	MOV M,A	1	A1	ANA C	1	CC	CZ a	3	F7	RST 6	1			
22	SHLD a	3	4D	MOV C,L	1	78	MOV A,B	1	A2	ANA D	1	CD	CALL a	3	F8	RM	1			
23	INX H	1	4E	MOV C,M	1	79	MOV A,C	1	A3	ANA E	1	CE	ACI d	2	F9	SPHL	1			
24	INR H	1	4F	MOV C,A	1	7A	MOV A,D	1	A4	ANA H	1	CF	RST 1	1	FA	JM a	3			
25	DCR H	1	50	MOV D,B	1	7B	MOV A,E	1	A5	ANA L	1	D0	RNC	1	FB	EI	1			
26	MVI H,d	2	51	MOV D,C	1	7C	MOV A,H	1	A6	ANA M	1	D1	POP D	1	FC	CM a	3			
27	DAA	1	52	MOV D,D	1	7D	MOV A,L	1	A7	ANA A	1	D2	JNC a	3	FD	---	-			
28	---	-	53	MOV D,E	1	7E	MOV A,M	1	A8	XRA B	1	D3	OUT d	2	FE	CPI d	2			
29	DAD H	1	54	MOV D,H	1	7F	MOV A,A	1	A9	XRA C	1	D4	CNC a	3	FF	RST 7	1			
2A	LHLD a	3	55	MOV D,L	1				AA	XRA D	1	D5	PUSH D	1						