VÝZKUMNÝ A ZKUŠEBNÍ LETECKÝ ÚSTAV

PRAHA 9 - LETŇANY

VÝZKUMNÝ A ZKUŠEBNÍ LETECKÝ ÚSTAV
PRAHA - LETŇANY

Číslo zprávy: V - 1227/75

Číslo úkolu: OR 6000

Druh zprávy:
výzkumná

Konto VZLÚ: 359.13

NÁVRH NA DOPLNĚNÍ HARDWARE POČÍTAČE EC 1021 OPERACEMI V POHYBLIVÉ ŘÁDOVÉ ČÁRCE

Odpovědní řešitelé úkolu:

J. Sokol V. Sedláček

Jan Sokol Vojtěch Sedláček

Vedoucí odboru 3500:

Ing. Oldřich Kropáč, CSc

Technický náměstek:

Ing. Bedřich Typlt

Datum vydání:
15.11.1974
20.5.1975 doplňky

Distributor:
VZLÚ - Odbyt
Praha 9 -Letňany

Stupeň utajení:
pouze pro služební potřebu

Listů: 49
Příloh: 4 tab., 2 obr.
Poř.číslo výtisku:

Souhrn:

Zpráva obsahuje ideový návrh na sestrojení jednotky pohyblivé čárky jakožto samostatného doplňkového zařízení k počítači EC 1021, která umožňuje, aby se na tomto počítači mohly ekonomicky realizovat i vědeckotechnické výpočty vyžadující většinu operací v pohyblivé řádové čárce. Ve zprávě je podrobně popsán zvolený způsob řešení této úlohy jak z hlediska technického řešení jednotky, tak z hlediska úpravy dosavadních mikroprogramů počítače EC 1021 a nově realizovaných 14 mikroprogramů v pohyblivé čárce. Studie je doplněna harmonogramem prací a přehledem očekávaných nákladů jakož i zhodnocením ekonomické efektivnosti celého záměru.

Rozdělovník:

1. Jan Sokol, Výzkumný ústav matematických strojů
   Stodůlecká ul., Praha 5 Jinonice
   vedoucí řešitelského týmu . . . . . 4 x

2. OTŘ GŘ Aero, s.inž.Jurásek . . . . . 2 x

3. VZLÚ 3500, řešitelské pracoviště . . . . . 5 x

O B S A H

1. VŠEOBECNÉ ÚDAJE

1.1. Formulace úlohy

Při přípravě nasazení počítače EC 1021 ve VZLÚ Praha - Letňany se ukázalo, že pro velkou část úloh, které se zde řeší, není tento počítač příliš vhodný. Jedná se hlavně o tzv. vědeckotechnické výpočty, zejména o úlohy z maticového počtu, kde se nepříznivě projevuje programová realizace aritmetiky v pohyblivé řádové čárce na počítači EC 1021. Tento počítač nebyl koncipován pro výpočty tohoto druhu a jednotlivé operace pohyblivé čárky (v dalším píšeme zkráceně PČ) jsou realizovány podprogramy. Tyto podprogramy byly sice ve VÚMS (a nezávisle na tom i v PVT Brno) vytvořeny v několika verzích a poměrně důkladně optimalizovány z hlediska rychlosti, nicméně je i současná verze ve srovnání s hardwarovou PČ jiných počítačů velmi pomalá. Tato okolnost není na závadu tam, kde se výpočty v PČ provádějí jen občas a v malých objemech; nejnevýhodněji se projevuje v úlohách iteračního a maticového charakteru, kde je počet operací v PČ vysoký. Jak ukázala porovnávací měření, provedená ve VZLÚ, převáží tato nevýhoda u typických úloh všechny ostatní výhody počítače EC 1021, který je právě při řešení úloh tohoto typu téměř ve všech případech značně pomalejší než např. MINSK 22 (viz tab.1.). Protože podprogramy PČ nelze už znatelně zrychlit, vznikl požadavek na vybavení počítače 1021 hardwarovou jednotkou pohyblivé čárky (JPČ).

Prvním krokem bylo zjistit, zda je tento požadavek prakticky řešitelný, tj. řešitelný při splnění dalších omezujících podmínek. Hlavní omezující podmínky pro řešení byly tyto:

a) malý počet změn do stávajícího hardware 1021,

b) termín uvedení do provozu nejdéle do poloviny roku 1976,

c) rozumný objem nákladů a zejména práce (asi 20 člověkoroků),
d) zvládnutelnost vlastními prostředky (dostupné součástky a technologie).

1.2. Přehled alternativních možností řešení

Počítač 1021 je mikroprogramovaný, na první pohled by se tedy nabízela možnost, vytvořit PČ čistě mikroprogramovou, bez zásahu nebo rozšiřování ostatního hardware počítače. Podrobnější rozbor však ukazuje, že

a) takto realizovaná PČ by nedosáhla požadované rychlosti
b) do hardware by stejně bylo třeba vestavět registry PČ
c) řídící paměť počítače je téměř plná, takže mikroprogramy PČ by se do stávajících kvádrů stejně nevešly.

Aby se na druhé straně předešlo nepříjemným zásahům a změnám v dosavadním hardwaru, přichází v úvahu pouze kombinované řešení: navrhnout a realizovat samostatnou aritmetickou jednotku pohyblivé čárky (JPČ) s vlastním řazením a v mikroprogramech provést pouze nezbytné doplnění instrukcí pro komunikaci s JPČ.

V dalším uvedeme některé alternativní možnosti řešení dílčích otázek a pokusíme se vytypovat schůdné kombinace alternativ.

1. Formát instrukcí

Úplná PČ vyšších modelů JSEP má celkem 44 instrukcí, odvozených od sedmi základních funkcí. Tento počet nelze v řídicí paměti JS1021 rozumně realizovat. Protože však zabudování nové PČ do software MOS znamená stejně výměnu interpretačních podprogramů za jiné (podstatně kratší) ve vyšších programovacích jazycích, kdežto u nově psaných programů

v Assembleru není formát instrukcí vůbec kritický, lze oprávněně formát instrukcí pozměnit tak, aby se s menším počtem instrukcí dosáhlo stejného výsledku. V zásadě jsou možná dvě řešení:

a) realizovat mikroprogramy 7 nových instrukcí (plnění registru, přičítání, odčítání, násobení, dělení, porovnání a uložení do paměti) a to buď ve dvou variantách (krátké a dlouhé, tj. s 4- a 8-bytovými operandami), nebo jen v dlouhé variantě, bude-li dostatečně rychlá.

b) realizovat mikroprogramem jen dvě (čtyři) "instrukce", totiž vysílací a přijímací, přičemž typ operace bude kódován v prvním byte vysílaných dat. Přijímací instrukce realizuje pouze funkci uložení do paměti.

2. Registry (střadače) PČ

Ve vyšších modelech JSEP jsou zabudovány čtyři osmibytové rychlé registry PČ, celkem tedy 32 byte. Možnosti řešení:

a) využít jako registry PČ vyšší univerzální registry stavu P4 (8 až F), které software nikdy nepoužívá

b) použít jen dva registry stavu P4 a vytvořit JPČ s jediným registrem,

c) zahrnout jeden nebo čtyři registry do JPČ a realizovat je tam obvodově.

3. Napojení JPČ k základní jednotce EC1021

a) vyvést na vhodném místě (např. na výstupním registru hl. paměti) nový bus, který by byl hradlován některou z nepoužívaných řídicích kombinací

b) využít k napojení bloku W, tj. bloku pro přímé spojení s jinou ZJ, včetně jeho ovládacích signálů beze změny v hardware JS 1021.

4. Výrobní realizace

a) v samostatné skříňce, připojené asi 40-žilovým kabelem

b) v neobsazených buňkách skříně ZJ.

5. Mikroprogramy ZJ

a) drobnými úpravami stávajících mikroprogramů vyšetřit místo pro potřebný počet nových "instrukcí" (přešití řádově po 50 mikroinstrukcích ve dvou až třech kvádrech, II, IN evt. IL)

b) využít stávajících instrukcí pro spojení mezi dvěma ZJ (instrukce RDD,WRD) - přešití cca 30-50 mikroinstrukcí

c) přešít celý kvádr II s mikroprogramem ZJ

Souhrnné zhodnocení jednotlivých alternativ

Z hlediska realizace považujeme za nejvhodnější volbu alternativ 1a, 2c, 3b, 4b a 5c. Lze také uvažovat o variantě 1b, 2b, 3b, 4b a 5b, která se zdá být nejméně pracná, přinese však podstatně horší výsledky a v některých bodech není ještě zcela vyjasněna. První uvedená varianta přinese vedle velmi slušných parametrů PČ navíc znatelné zrychlení počítače i v dosavadních instrukcích. Při volbě 5c lze očekávat ve váženém průměru zrychlení počítače jako celku zhruba o 25%, v některých důležitých a často používaných rutinách např. aupervizoru až o 80 %, tj. z dosavadních cca 25 tis. instrukcí na 32 až 45 tisíc.

1.3. Diskuse zvoleného řešení

V tomto odstavci uvedeme podrobnější diskusi zvoleného řešení ve vztahu k požadovanému cíli, vedlejším podmínkám jakož i k ostatním možným alternativním řešením.

Protože základní jednotka 1021 je mikroprogramovaná, bylo nejprve třeba zvážit možnost čistě mikroprogramového řešení bez zásahu ostatního hardware. Důkladným rozborem stávajících mikro-

programů bylo zjištěno, že do dosavadních šesti kvadrantů paměti mikroprogramů (ROS) se mikroprogramy PČ v žádném případě nevejdou. Další práce však ukázala, že ani přidáním dalšího kvadrantu, které by bylo v principu možné, by se úloha nevyřešila příliš dobře; mikroprogramové operace PČ na dosavadní aritmetické jednotce 1021 vycházely stále ještě poměrně pomalé: slučování řadově 200usec, násobení kolem 1msec. Proto jsme se rozhodli pro řešení obvodové, tj. vytvoření více méně nezávislé jednotky pohyblivé čárky, která bude mikroprogramem pouze ovládána jako celek.

Dalším problémem byla otázka připojení této obvodové JPČ k dosavadnímu hardware, jejího ovládání mikroprogramem a návrh logického interface mezi oběma celky. Rezervní bity v mikroinstrukci počítače dávají sice možnost vestavět pro JPČ zvláštní samostatné připojení s novými hradly mezi JPČ a hlavní pamětí (registr M0-M3), abychom však minimalizovali nutné zásahy do stávajícího hardware a nebránili případnému využití rezervních bitů pro jiné účely, rozhodli jsme se využít pro připojení JPČ blok W, původně určený pro spojení mezi dvěma počítači. Hradla i ovládací signály tohoto bloku pro náš účel plně vyhovují a ztráta možnosti propojení více počítačů nebyla pro VZLÚ (a patrně i pro ostatní uživatele) kritická. Zatím nedořešena zůstala otázka zabezpečení této spojové cesty, neboť blok W nemá paritu.

Logický interface JPČ, čili formát předávaných dat, byl ovlivněn úvahami o konstrukci JPČ, jednoduchosti mikroprogramů a skutečnými požadavky programů. Vzhledem k předpokládané rychlosti vlastní JPČ jsme se rozhodli pro jediný formát, který je tvořen jednobytovým "operačním znakem" pro JPČ a osmibytovým číslem - operandem. Z hlediska mikroprogramu se tedy JPČ jeví jako

jednoadresový počítač se sedmi zakladními operacemi plnění registru, uložení registru, sčítání, odčítání, porovnání, násobení a dělení. Spojení počítače 1021 s JPČ se zahajuje vysláním "operačního znaku", který určuje druh operace a číslo registru. Potom vysílá ZJ do JPČ (u operace uložení registru obráceně JPČ do ZJ) osmibytový operand ve tvaru dvojnásobného čísla v pohyblivé čárce podle JSEP, kde první byte obsahuje znaménko čísla a šestnáctkovou charakteristiku, zbývajících sedm byte tvoří mantissu v přímém kódu. S výjimkou operace uložení registru vyčká ZJ dokončení operace v JPČ a příjme podmínkový byte, v němž je zakódován podmínkový kód (cc) výsledku operace, případně údaj o přetečení exponentu.

Rozhodnutí zahrnout čtyři registry pohyblivé čárky do JPČ místo využít pro ně nepotřebných registrů stavu P4 v zápisníku počítače bylo dáno požadavkem kompatibility s počítači 1021 bez JPČ a požadavkem rychlosti operací. Naproti tomu rychlost aritmetiky JPČ (vlastní čas slučování max.10 usec na 8 byte) dovoluje převádět čtyřbytové operandy "krátkých" instrukcí mikroprogramem na standardní osmibytový tvar prostým přidáním čtyř nulových byte. Rozdíl v rychlosti je nepatrný v poměru k trvání základního cyklu operace a JPČ se tím zjednoduší. Vnitřní sdílení času mezi JPČ a ZJ by mělo smysl u dlouhých operací (násobení a dělení), kde je však komplikováno tím že ZJ musí reagovat na signály přetečení a dělení nulou. Proto bylo zatím ponecháno stranou.

Požadavek jednoduchosti autonomního řadiče JPČ i příslušných mikroprogramů vedl k rozhodnutí omezit oparační kód pohyblivé čárky oproti JSEP. Rozborem dosavadního software počítače 1021, pokud pracuje s pohyblivou čárkou, a zejména překladače FORTRAN bylo zjištěno, že lze zcela vynechat operace nenormalizovaného slučování a všechny operace formátu RR (registr-registr). Operace RR budou simulovány makroinstrukcemi, které je převedou

na dvojice operací formátu RX. Nenormalizované operace mají význam pouze pro speciální výzkumy v oblasti numerické stability apod., které se ve VZLÚ nepředpokládají. V souvislosti s tím lze také odbourat dvojí způsob reakce na přetečení exponentu, řízený bitem v uživatelské masce. Tato možnost se běžně nevyužívá a programátor např. ve FORTRANu k ní vůbec ani nemá přístup. Až na tyto výjimky bude JPČ plně kompatibilní s počítači JSEP - tedy podstatně víc, než dosud používané podprogramy.

Počítač 1021, vybavený JPČ podle tohoto návrhu, bude tedy pracovat s čísly v pohyblivé čárce ve tvaru podle JSEP o délce 4 nebo 8 byte a bude obsahovat navíc 14 operací, které rovněž přesně odpovídají JSEP. Jsou to operace se symbolickými znaky LE, LD, STE, STD, AE, AD, SE, SD, CE, CD, ME, MD, DE, DD, viz tab.2. Strojový tvar instrukcí, podmínkový kód, adresy registrů a reakce na přetečení a dělení nulou odpovídají rovněž JSEP. Předpokládané rychlosti těchto operací jsou uvedeny v tabulce 3.

2. NÁVRH TECHNICKÉHO ŘEŠENÍ JEDNOTKY POHYBLIVÉ ČÁRKY (JPČ)

2.1. Úvod

Jednotka pohyblivé čárky (dále jen JPČ) je řešena jako samostatně připojitelná část k základní jednotce počítače EC 1021 (dále jen ZJ). Korespondence mezi JPČ a ZJ z hlediska povelových signálů je popsána v samostatné kapitole 2.4. "Styk JPČ se ZJ". Datový styk JPČ se ZJ je definován následujícím způsobem:

2.1.1. Instrukční byte - V osmi bitech tohoto byte jsou sdělovány základní jednotkou bloku JPČ dvě základní informace.

a) Instrukční kód - tato část obsahuje jednu ze sedmi určených kombinací, které odpovídá jedné ze sedmi požadovaných operací. Jsou to tyto operace:

LD - plnění vnitřní paměti JPČ
STD - uložení z vnitřní paměti JPČ
AD - přičtení k obsahu vnitřní paměti JPČ
SD - odečtení od obsahu vnitřní paměti JPČ
MD - násobení z obsahem vnitřní paměti JPČ
DD - dělení obsahu vnitřní paměti JPČ
CD - porovnání s obsahem vnitřní paměti JPČ

b) Adresa registru - v této části instrukčního byte je zakódovaná kombinace označující jednu ze čtyř pamětí operandů v JPČ. Přesto, že s uživatelského (programátorského) pohledu se jedná o čtyři registry pohyblivé aritmetiky, nazýváme je pamětí operandů, neboť jsou jako paměti technicky realizovány.

2.1.2. Operand - Operandem rozumíme vždy takto uspořádanou osmici bytů. V pořadí první vstupující (resp. vystupující) byte obsahuje jeden bit znaménka mantisy a sedm bitů charakteristiky. Charakteristika je v přímém kódu, hexadecimálně a je zvětšena o 64.

V pořadí druhý byte obsahuje nejvýznamnější bity mantisy, osmý byte pak obsahuje nejnevýznamnější bity mantisy. Celá mantisa je tvořena 56 bity. Vstupuje i vystupuje vždy v přímém binárním kódu.

2.1.3. Podmínkový byte - Tento byte sděluje JPČ do ZJ u příslušných instrukcí (jak bude dále vysvětleno) výsledky těchto testů:

a) Podmínkový kód - vypovídá o tom, zda výsledný operand je větší než nula, menší než nula, nebo roven nule
b) Příznak přetečení charakteristiky ke které došlo při operaci.
c) Příznak dělení nulou.

2.2. Blokové schéma JPČ

Z rozborů požadavků na rychlost, spolehlivost a technickou realizovatelnost JPČ jsme se rozhodli pro následující pojetí řešení JPČ, viz obr.1. JPČ sestává z těchto funkčních bloků:

a) PŘENOSOVÝ BLOK (PB)
b) PAMĚŤ (M)
c) JEDNOTKA REGISTRU (RJ)
d) ARITMETICKÁ JEDNOTKA (AJ)
e) ŘADIČ (Ř)

Nyní se zmíníme o funkčních blocích podrobněji.

2.2.1. PŘENOSOVÝ BLOK - Obstarává veškerý datový přenos mezi JPČ a ZJ v obou směrech. Od ZJ přijímá instrukční byte a vstupující operand. Každý vstupující byte je kontrolován na správnost parity v obvodu TP (TEST PARITY) a ve vstupním registru (RI) je převeden z paralelně vstupující osmice bitů na dvě seriově vystupující čtveřice bitů. Přenosový blok do ZJ pak předává podmínkový byte a vystupující operand. Přitom převádí ve výstupním registru (RO)

čtveřice bitů do bytovéstruktury (dvě čtveřice seriově je jeden paralelní byte). Vystupující byte je opatřen bitem příčné parity v generátoru parity (GP).

2.2.2. PAMĚŤ (M): Paměť slouží k uchování čtyř operandů ve čtyřech programátorsky přístupných "Registrech". Jako nejvýhodnější pro daný účel bylo zvoleno následující uspořádání. Mantisy jsou uloženy v pamětech s přímým přístupem - paměti mantis (MM1-MM4). Jsou realizovány na čtyřech 64 bitových RAM typu SN 7489. Charakteristiky jsou uloženy odděleně ve čtyřech osmibitových pamětech - paměti charakteristik operandu B(MEB1-MEB4). Znaménka mantis uložených operandů jsou uložena ve čtyřech jednobitových pamětech - pamětech znamének (MS1 až MS4). Toto umožňuje nezávislý přístup k charakteristikám, mantisám a znaménku. Tento blok obsahuje také obvod pro kontrolu nulovosti vstupující mantisy - test nulovosti mantis (TØM). Výsledek testu je zapsán do jedné ze čtyř jednobitových pamětí - paměti testu nulovosti mantisy (MT1 až MT4). Zmíněné paměti (tj. MM1-MM4, MS1 až MS4, MT1 až MT4, MEB1 až MEB4) jsou společně adresovány, přičemž adresa je určena instrukčním byte a po celou dobu průběhu operace JPČ se nemění. Součástí bloku je osmibitová paměť vstupujícího operandu charakteristiky - paměť charakteristiky operandu A (MEA). V bloku jsou též dvě jednobitové paměti znamének mantis operandů se kterými se provádí aritmetická operace - pamět znaménka operandu A(MSA) a paměť znaménka operandu B(MSB). Toto uspořádání umožňuje nezávislou generaci znaménka výsledku bez ohledu na probíhající operaci. Tuto činnost provádí generátor znamének (GS).

2.2.3. JEDNOTKA REGISTRU - Tato jednotka obsahuje jeden univerzální serioparalelní registr - operační registr (X). Umožňuje 4 bitový paralelní obousměrný posuv (t.zn.je možné do něj vstupovat i vystupovat zleva i zprava), čím je usnadněno a zrychleno srovnávání charakteristik a normalizace. Obousměrný vstup i výstup do tohoto registru je vhodný při změně pořadí bitů mantisy operandu. Pro vlastní algoritmy násobení a dělení je vybaven možností obousměrných posuvů mantisy operandů pouze o jeden bit. Může tudíž obousměrně posouvat ryze seriově. Vzhledem k jeho komplikovanosti a vzhledem k požadavku na jeho pracovní rychlost, rozhodli jsme se pro jeho realizaci použít IO zahraniční výroby SN 74194. Druhý operační registr v tomto bloku - operační registr (Y) je opět serioparalelní (4bity paralelně, 16 bitů seriově) s možností jednosměrného vstupu zleva a jednosměrného posuvu o 4 bity vpravo. Přesto, že tento registr není již tak komplikovaný jako registr X, je zvolen s hlediska spolehlivosti a realizovatelnosti IO SN 7491A. Dalším pracovním registrem bloku je osmibitový jednosměrný registr pro práci s charakteristikami - operační registr charakteristik (E). Funkčně by bylo možné potřebné operace s charakteristikami v operačních registrech X a Y provádět, ovšem za cenu podstatného prodloužení vlastního úkonu (bylo by třeba uvolnit event. mantisy z operačních registrů X, Y, a zároveň v podstatně více krocích provést vlastní úkol z charakteristikami).

Proto je organicky do tohoto bloku včleněn reverzibilní osmibitový čítač s možností paralelního zápisu a čtení - - reverzibilní čítač charakteristik (CE). Splňuje dvě funkce. Jednak při normalizaci přičítá (event.odčítá) tak, jak vyžaduje změna charakteristiky. Jednak umožňuje jednoduchým odčítáním

do nulového stavu určit potřebný počet hodinových impulsů nutných pro posuv mantisy při srovnávání charakteristik u operací slučování. Pro svoji značnou komplikovanost je realizován obvodem SN74193.

2.2.4. ARITMETICKÁ JEDNOTKA (AJ) - Tato jednotka provádí na paralelních čtyřech bitech následující úkony:

a) generuje na výstupech čtyři nuly
b) generuje na výstupech čtyři jedničky
c) převádí operand v přímém tvaru z R vstupu na výstup
d) převádí operand v přímém tvaru z S vstupu na výstup
e) invertuje operand na vstupu R do výstupu
f) invertuje operand ze vstupu S do výstupu
g) binárně sečítá operand ze vstupu R s operandem ze vstupu S na výstup
h) binárně sečítá přímý operand R a inverzní operand S
j) binárně sečítá inverzní operand R a přímý operand S
k) sečítá inverzní operand R a inverzní operand S na výstup
l) přičítá k operandu R jedničku
m) přičítá k operandu S jedničku
n) přičítá k výsledku jedničku

Tyto úkony provádí čtyřbitová úplná sčítačka (S) IO typ SN 7483 A ve spojení se dvěma obvody pro tvorbu doplňku (D1,D2). Tyto obvody jsou realizovány pomocí IO SN 74H87. Zvolená konfigurace umožňuje elegantním způsobem vyhovět požadovaným nárokům na vlastnosti eritmetické jednotky při zachování vysoké operační rychlosti a vysoké spolehlivosti. Vzhledem k tomu, že je nezbytné operační registry připojit k jednotlivým vstupům AJ, rozhodli jsme se pro nejjednodušší možné řešení, tj. opatřit každý datový vstup samostatným obvodem pro tvorbu doplňku.

Blok AJ dále vyžaduje jednobitovou paměť přenosu - obvod přenosu aritmetické jednotky (CR Y).

2.2.5. ŘADIČ: Pomocí ovládacích signálů řídí činnost výše popsaných bloků. Cvládací signály jsou dvojího typu. Jednak se jedná o statické signály (logické úrovně) uchovávané po celou dobu řízeného úkonu, jednak dynamické (clock), které taktují činnost řízených bloků v průběhu úkonu. Vzhledem k tomu, že realizace řadiče jako paměti mikroinstrukcí by vedla k neúměrným nárokům na její kapacitu a nepříznivě ovlivnila rychlost vlastních operací, uvažujeme provést řadičové obvody pomocí rozdělovacích čítačů. V tomto konceptu má každá ze sedmi instrukcí svůj vlastní řídící čítač, jehož stavy určují sled makroinstrukcí a event. mikroinstrukcí. Mikroinstrukcí zde rozumíme paralelní provedení několika kroků, makroinstrukcí pak uzavřený sled mikroinstrukcí. Makroinstrukce jsou též realizovány čítači. Součástí řadiče je testovací obvod násobitele (TMD), testující vždy nejméně významný bit mantisy násobitele, jak to vyžaduje řízení vlastní operace násobení daným algoritmem. Dalším obvodem řadiče je generátor výsledného bitu mantisy podílu - generátor podílu (GDD) používaný při operaci dělení. Řadič zahrnuje také generátor podmínkového byte (GPB)

2.2.6. Struktura přenosu dat v JPČ.

Přenos dat v rámci JPČ probíhá zásadně vždy po čtyřech bitech paralelně. Abychom jednoduchým způsobem vyřešili datové sdílení mezi jednotlivými bloky bez použití přídavných multiplexorů, použili jsme tří datových sběrnic (bus). Bus je v tzv. Three-State-Logic. Pro tento účel jsou s výhodou použity speciální IO SN 74126, kterými lze ovládáním vstupu inhibit odpojit

zcela vstupující data od sběrnice. Z hlediska korespondence dat v JPČ vyhovuje následující uspořádání bus(sběrnic).

R-BUS - je registrovou sběrnicí, na níž lze vstoupit daty z operačního registru X(výstupy registru zleva i zprava), z registru operačního charakteristiky (E), z reverzibilního čítače charakteristik (CE), z paměti charakteristiky operandu A(MEA). Výstup sběrnice je připojen na operandový vstup R aritmetické jednotky. Tato sběrnice je typu "Three-state-logic".

S-BUS - Je sběrnicí pamětí, na kterou je připojen výstup z operačního registru Y, výstup z přenosového bloku (PB), výstup z generátoru znaménka (GS), výstup z paměti charakteristiky operandu B(MEB), výstupy z pamětí znamének (MS), výstup z paměti testu nulovosti mantisy (MT). Výstup paměti mantis (MM). Protože použité RAM v MM mají inverzní výstup s otevřeným kolektorem, jsou na S-bus připojeny negujícím invertorem (I), vyhovujícím funkci inhibit ve smyslu definice S-bus. Dále je na tuto sběrnici připojen výstup z generátoru podmínkového byte (GPB) a výstup z generátoru podílu (GDD). Tato sběrnice je typu "Three-State-Logic".

T-BUS - Tato sběrnice je výstupní sběrnicí aritmetické jednotky. Je na ní připojen vstup výstupního registru (RO), vstup do operačního registru X zleva, vstup do operačního registru Y (zleva), vstup do operačního registru charakteristiky E, vstup do reverzibilního čítače charakteristiky (CE), vstup do pamětí znamének (MSA, MSB, MS1 až MS4), vstup do paměti charakteristiky operandu A(MEA), vstup do paměti charakteristiky operandu B(MEB1

až MEB4), vstupy do pamětí mantis (MM1 až MM4) a vstup do obvodu testu nulovosti mantis (TØM). Protože na tuto sběrnici je připojen pouze jediný výstup (z aritmetické jednotky), je provedena klasickým způsobem.

Zvláštním vstupem dat je přímý přístup k operačnímu registru X. Tento vstup je zprava. Toto opatření je nutné, neboť při zápisu mantisy, která je vždy v obráceném pořadí (od nejvýznamnějšího bitu počínaje) tímto dochází k právnému uspořádání pro aritmetické operace a zároveň se nezatěžují hlavní sběrnice, které mohou tudíž přenášet data současně při plnění registru X např. z paměti mantis do operačního registru Y apod.

2.3. Algoritmizace operací JPČ

V této kapitole pojednáme o způsobu vlastního provádění jednotlivých operací.Vstupní operand (vstupující ze ZJ) budeme označovat jako op.A, uložený operand (ve vnitřní paměti JPČ) budeme označovat operandem B.

2.3.1. LD - Operace je určena k přenosu jednoho operandu definovaného v kap.1.2 ze ZJ do jedné ze čtyř vnitřních pamětí JPČ. Nejprve je zapsáno znaménko do paměti znaménka operandu A. Část charakteristiky (jako bylo uvedeno je v JPČ charakteristika přesouvána stejně jako ostatní datové informace paralelně po čtveřicích bitů) je vložena ve stejné době do paměti charakteristiky operandu A, reverzibilního čítače charakteristik a do operačního registru charakteristiky. V následujícím kroku je přesunuta stejným způsobem druhá část charakteristiky operandu A.

V této době zároveň asynchronně probíhá nulování operačního

registru X. Poté je přicházející mantisa operandu A ze ZJ, zapisována do operačního registru X zprava, čímž je obráceno pořadí významnosti bitů mantisy. Protože operand A nemusí přicházet nutně v normalizovaném tvaru, proběhne vlastní normalizace, o které se zmíníme podrobněji a v následujících popisech se na ní budeme odvolávat.

Princip normalizace spočívá v upravení mantisy posouváním po čtveřicích bitů doleva, tzn. k nevýznamnějším bitům, přičemž s každým posuvem je odečteno od charakteristiky po jedné. Vlastní provedení vypadá takto. Při vstupu mantisy operandu A do operačního registru X se provádí současně test na její nulovost. Je-li mantisa nulová, nemá totiž normalizace smysl a neprovádí se. V opačném případě je testována nejvyšší čtveřice bitů mantisy operandu A (14.čtveřice) na přítomnost alespoň jednoho nenulového bitu. Testovací obvod je součástí operačního registru.X. Je-li v této čtveřici alespoň jeden bit nenulový, operand je pokládán za normalizovaný. Jinak je obsah operačního registru X posunut o jednu čtveřici bitů vlevo a zprava je doplněn nulami. Současně je charakteristika, uložená v reverzibilním čítači charakteristik, zmenšena o jednu. Stav tohoto čítače je trvale testován na případné přetečení (podtečení). Tento cyklus proběhne tolikrát, kolikrát je potřeba tzn. až obvod testu normalizace zjistí přítomnost alespoň jednoho nenulového bitu v poslední (nejvyšší) čtveřici operačního registru X. Protože mantisa vstupující do JPČ je vždy v přímém tvaru a pro vlastní činnost JPČ je výhodnější pracovat s jednotkovými doplňky, je mantisa při přepisu z registru X (již znormalizovaná) do příslušné paměti aritmetickými obvody případně invertována. Výsledná charakteristika je z reverzibilního čítače charak-

teristik přesunuta do paměti charakteristiky operandu B, současně je vložen znaménkový bit z pamětí znaménka operandu A do příslušné paměti znamének operandu B. Adresový výběr pamětí mantis, charakteristik a znamének je samozřejmě, jak bylo již uvedeno, v této i ve všech ostatních operacích nezměněn.

2.3.2. STD - Tato operace je určena k přenosu operandu z paměti JPČ do ZJ. Vzhledem k tomu, že vystupující operand musí býti uspořádán stejně jako vstupující operand, je nejprve vybrán znaménkový bit z paměti znamének a charakteristika z paměti charakteristik operandu B. Samozřejmě se opět charakteristika převádí ve dvou taktech. Znaménko mantisy, které při výběru bylo vloženo do paměti znaménka operandu A, určuje, zda mantisa vystupujícího operandu bude ponechána (kladné znaménko) nebo invertována (záporné znaménko). Charakteristika je uložena do výstupního registru přenosového bloku. Aby bylo vyhověno úvodnímu předpokladu, je mantisa přesunuta do operačního registru, tak, že se z paměti mantis operandu B vybírá v obráceném pořadí. Mantisa prochází aritmetickou jednotkou, kde je event.invertována a zapsána do výstupního registru přenosového bloku.

2.3.3. AD, SD - Tyto operace jsou určeny k slučování. AD přičítá operand A k operandu B, součet je uložen na místo operandu B. SD odčítá operand A od operandu B a výsledek je opět uložen na místo operandu B. Protože algoritmus obou operací je v podstatě stejný, popisujeme obě operace společně. Vstupující znaménko a charakteristika operandu A je stejně jako u instrukce LD zapsána do příslušných pamětí. Tzn. znaménko do paměti znaménka operandu A a charakteristika do paměti charakteristiky operandu A a zároveň do reverzibilního čítače charakteristik.

Vstupující mantisa je zapisována do operačního registru X zprava (změna pořadí významnosti bitů) a současně mantisa operandu B do operačního registru Y zleva. Operand A jest nyní normalizován standardním způsobem, tj. jak bylo popsáno u operace LD. V těchto operacích je nezbytné, aby slučované operandy byly srovnány dle charakteristik. Z tohoto důvodu se nejprve charakteristiky obou operandů v aritmetické jednotce odečtou. Znaménko rozdílu určuje mantisu operandu, která musí býti posunuta po čtyřech bitech vpravo tolikrát, kolikrát je určeno absolutní hodnotou rozdílu charakteristik. Mantisa operandu A je dle znaménka mantisy případně invertována, neboť tento operand byl zapsán v přímém tvaru. Vlastní inverze je provedena současně s operací slučování. Vlastní sčítání probíhá postupně po čtveřicích bitů. Operand A vystupuje z operačního registru -X od nejnižších bitů podobně jako operand B z operačního registru Y do aritmetické jednotky. Výsledek je zapisován zleva do operačního registru X. Dojde-li po úplném sloučení obou operandů (16 clock) ke kruhovému přenosu jedničky, je k obsahu operačního registru X novým průchodem aritmetickou jednotkou přičten jeden bit. Výsledek je opět vložen do operačního registru X. Při tomto úkonu je přičítaná jednička generována aritmetickou jednotkou, vstup z operačního registru Y je tudíž uzavřen. Nastalo-li při slučování přetečení mantisy (o jeden bit) je charakteristika v reverzibilním čítači charakteristik zvětšena o jednu a zároveň se obsah operačního registru X posune o jednu čtveřici doprava. Zleva vstupující bity mají hodnotu znaménka. Před zapsáním výsledku operace do vnitřních pamětí operandů se v případě, že nedošlo k přetečení mantisy, provede normalizace standardním způsobem. Poté je mantisa bez úprav uložena do příslušné paměti

mantis a současně charakteristika do paměti charakteristik operandu B a znaménko do pamětí znaménka operandu B. Průběžně je také dle znaménka mantisy ukládaného operandu testována na nulovost. Test je uchován v příslušné paměti nulovosti operandu B. Po skončení vlastní operace je generován podmínkový byte a je předán výstupním registrem přenosového bloku ZJ.

2.3.4. MD - Tato operace je určena k násobení vstupujícího operandu A s určeným uloženým operandem B. Výsledek operace (součin) je uložen na místo operandu B. V podmínkovém byte je po skončení operace ZJ sdělováno event. přetečení charakteristiky součinu.

Vstupující charakteristika a znaménko mantisy jsou zapsány do pamětí charakteristiky operandu A a paměti zmanénka operandu A. Charakteristika je současně zapsána do reverzibilního čítače charakteristik a operačního registru charakteristik. Dalším krokem je vloženo znaménko mantisy operandu B do paměti znaménka operandu B, tím je umožněno nezávisle na dalších krocích generovat znaménko výsledku. Mantisa operandu A, která se současně uložila v náležitém smyslu (zprava) do operačního registru X, je přepisována do operačního registru Y. Při tomto přepisu je zachován přímý kód mantisy operandu A. Operační registr X, do kterého bude v průběhu operace střádán dílčí výsledek násobení je vynulován. Zároveň s tímto úkonem je operační registr Y kruhově uzavřen. Tak je uchována mantisa operandu B v průběhu operace násobení. Z operačního registru charakteristiky je vybrána charakteristika A, z paměti charakteristik operandu B je vybrána příslušná charakteristika operandu B a oba tyto exponenty jsou aritmetickou jednotkou sečteny.

Výsledný součet charakteristik je uložen do reverzibilního čítače charakteristik. Vlastní operace násobení proběhne až do konce i v případě že nastalo přetečení charakteristiky, neboť tato skutečnost bude standardně sdělována v podmínkovém byte. Následuje 56x opakující se cyklus, při kterém se dle čítače bitů (v řadiči),nastaveném do počátečního stavu (nulového) testuje v pořadí odpovídající bit násobitele. Výsledek testu určuje, zda k sečtení obsahu operačního registru X s obsahem operačního registru Y dojde či nikoliv. Výsledek dílčího součtu je ukládán do registru X a posunut o jeden bit vpravo. Po skončení posledního dílčího součtu je součin uložený v operačním registru X testován na přetečení mantisy. V případě přetečení se obsah operačního registru X posune o jednu čtveřici bitů vpravo se vstupujícími nulovými bity zleva a současně je stav reverzibilního čítače exponentů zvětšen o jedničku. Stav čítače je samozřejmě opět testován na přetečení. Výsledek je dále, je-li třeba, standardním způsobem normalizován a poté uložen případně v inverzním tvaru (dle znaménka výsledku z generátoru znamének) do paměti mantisy, charakteristiky a znaménka operandu B. Během tohoto úkonu je vysílán do ZJ podmínkový byte.

DD -)
2.3.5./ Tato operace je určena k dělení operandu B uloženého v paměti JPČ vstupujícím operandem A. Výsledek (podíl) je uložen na místo vybraného operandu B. Znaménko a charakteristika operandu A je standardně zapsána do paměti charakteristiky a znaménka operandu A. Mantisa operandu A je vkládána do registru X zprava. Současně probíhá test nulovosti mantisy operandu A. Operand A je v operačním registru X standardním způsobem normalizován, za předpokladu, že mantisa tohoto operandu není nulová. V případě její nulovosti je poslán do základní jednotky podmínkový byte s příznakem této skutečnosti a operace je ukončena.

V následujícím kroku je v aritmetické jednotce odečtena charakteristika operandu A od charakteristiky operandu B. Výsledek je zapsán do reverzibilního čítače charakteristik. Příznak event. přetečení charakteristiky je zapsán do paměti podmínkového byte. Nyní je mentisa, uložená v operačním registru X, přepsána do operačního registru Y, tak, že z registru X vystupuje zprava. Dále je vybrána z pamčti mantisy operandu B mantisa, která se případně v aritmetické jednotce invertuje podle jejího znaménka. Z aritmetické jednotky vstupuje do operačního registru X zleva. V následujícím kroku je v aritmetické jednotce odečtena mantisa v registru Y od mantisy uložené v operačním registru X. Podle znaménka rozdílu se generuje výsledný bit podílu tak, že kladnému znaménku odpovídá jednička a zápornému nula. Zároveň je tím určeno, zda bude následovat přičtení nebo odečtení mantisy registru Y v následujícím taktu. Po každém jednom z těchto úkonů je registr posunut o jeden bit vlevo, přičemž je zprava doplňován nulami. Tento cyklus je vykonán 56krát, tzn. v celém rozsahu mantisy. Podíl z generátoru podílu je ukládán v obráceném pořadí do odpovídající paměti mantisy, původně obsazené dělencem; po naplnění této paměti je její obsah přepsán do registru X zleva a je provedena standardní normalizace. Po této normalizaci je dle vygenerovaného znaménka obsah operačního registru X případně invertován a uložen do paměti mantis operandu B. Současně je zapsána charakteristika, znaménko a příznak nulovosti do odpovídajících pamětí, stejně jako příznak přetečení charakteristiky do paměti podmínkového byte. Vysláním podmínkového byte do ZJ je tato operace ukončena.

2.3.6. CD - Tato operace je určena k porovnání operandu A s operandem B uloženým v paměti JPČ. V zásadě je tato operace stejná jako operace odčítání - SD, s tím rozdílem, že výsledek operace (rozdíl) není ukládán do paměti JPČ, ale je zapomenut. Znamená to, že porovnávaný operand B uložený v paměti JPČ zůstává nezměnen, Výsledek operace je v podmínkovém byte sdělen základní jednotce. Je v něm zakódován příznak porovnání: Operand A je větší než operand B, Operand A je menší než operand B, operand A je roven operandu B.

2.4. Styk JPČ se ZJ počítače EC 1021

Ve snaze omezit případné zásahy do zapojení ZJ počítače EC 1021 by bylo vhodné použít pro vlastní styk z JPČ informační vedení a řídící signály bloku W(MØ/.,W/.,MØW,WM3,atd), který je součástí ZJ. Jelikož blok W nepracuje s paritním bitem, bylo by nutné jednak rozšířit informační vedení o jeden bit (pro zajištění kontroly přenášených dat) provedením signálu MO/P z MO do W a signálu W/P z W do vstupních hradel M, jednak rozšířit W blok o obvody paritního bitu. Protože zpracování řídicích signálů MOW, WM3 a START by bylo pro použití JPČ jiné, než jak je provedeno v bloku W, byla by deska W.00.051 nahrazena deskou s jinou řídící částí a s registrem zvětšeným o paritní bit. Dále ve vstupních hradlech M by byla deska KWP.00.078 nahrazena deskou obsahující jednak hradlo partiního bitu W/P a jednak by na ní bylo zabráněno (na rozdíl od KWP) vytvoření parity z W/.. Umožnění vlastního styku JPČ se ZJ by bylo tudíž možné přidáním dvou spojů a výměnou dvou desek, z nichž jedna při běžné činnosti počítače nepracuje. (Běžnou činností rozumíme používání počítače bez součinnosti s jiným

počítačem). V případě potřeby je možno vysunutím nové desky na pozici W odpojit JPČ od ZJ. Činnost ostatních obvodů tím nebude nijak narušena.

2.5. Systém kontrol

Kontrola činnosti JPČ byla důsledně přizpůsobena úrovni kontrol u počítače EC 1021. Znamená to, že vstupní informace do JPČ je kontrolována příčnou paritou, výkonné obvody JPČ jsou zdvojeny pro paralelní zpracování informace v pozitivní i negativní logice a na výstupu JPČ je kontrolována správnost převodu z vnitřního kódu JPČ na kód ZJ. Výpovědí o bezchybném ukončení činnosti JPČ je vytvoření signálu "hotovo JPČ". Nutnou podmínkou pro generování tohoto signálu je správnost vstupní informace kontrolované paritou, správnost převodu vnitřního kódu na vnější, resp. obráceně, správnost činnosti vlastního řadiče JPČ, správnost činnosti operačních registrů JPČ a správnost činnosti aritmetické jednotky JPČ. Aby bylo možné kontrolovat činnost JPČ bez zásahu do zapojení počítače, uvažujeme o přivedení signálu "hotovo JPČ" do bloku L v ZJ (využity signály WY/P,WY/7). Blok L by v tomto případě byl testován po celou dobu práce JPČ. V případě, že nebude splněna alespoň jedna z podmínek pro vytvoření signálu "hotovo JPČ", bude tato okolnost indikována na panelu technika. Na žárovkové desce v bloku JPČ budou pak signalizovány relevantní stavy JPČ, mající vliv na vytvoření signálu "hotovo JPČ" a umožňující bližší lokalizaci místa závady JPČ. Je možno uvažovat ještě jiný způsob uplatnění signálu "hotovo JPČ" než je jeho zavedení do bloku L, např. do řídící smyčky ZM, což by si vyžádalo zásah do řídící smyčky ZM.

2.6. Konstrukční uspořádání a napájení.

Buňka z vlastními deskami JPČ bude umístěna na volné pozici B2A skříně OJPJ, čili v prostoru nad buňkou zápisníkové paměti. Napájení bude provedeno se samostatného primárního zdroje 8,5 V/10A. přes standardní sekundární stabilizátory, které budou umístěny přímo v buňce. Zdroje budou zapojeny na hlídací a měřící obvody zdrojové soustavy skříně OJPJ.

3. MIKROPROGRAMY

V popsaném rámcovém řešení nejsou požadavky na úpravu mikroprogramů příliš velké. Předně je třeba rozšířit soubor přípustných operačních znaků o dalších 14 operací formátu RX; za druhé je třeba vytvořit vlastní mikroprogramy těchto operací, které budou celkem jednoduché; a konečně je třeba pro tyto úpravy vyšetřit místo v paměti ROS. Na první pohled je zřejmé, že tyto úpravy nelze realizovat dodatečným přešitím příslušných adres, neboť počet dovolených oprav v jednom kvadrantu je asi 30. Nejmenší možnou úpravou tedy bude úplné přešití jednoho kvadrantu, tj. 512 adres paměti ROS. Protože první z požadovaných úprav musí být umístěna v kvadrantu II (kvadrant s mikroprogramem základní jednotky Z), zaměřili jsme se na možnost přeprogramování tohoto kvadrantu. Už v první etapě se ukázalo, že změnou algoritmu mikroprogramu Z a účelnějším naprogramováním lze ušetřit dostatečný počet adres a že tedy stačí, zaměřit se na přepsání tohoto jednoho kvadrantu s tím, že v nutných případech může být několik adres v jiných kvadrantech změněno dostatečným přešitím na místě. Zároveň se ukázalo, že vedle úspory místa bude možno mikroprogram Z zkrátit i dynamicky, takže se navrhovanou úpravou zrychlí i provádění ostatních operací počítače 1021.

Úprava mikroprogramů se tedy po prvé etapě rozpadla na čtyři dílčí problémy, z nichž prvé dva je nutno vyřešit v každém případě, další dva jsou lákavé s hlediska zvýšení parametrů stroje, nesouvisí však nutně s realizací JPČ. V dalším popíšeme zhruba navržené algoritmy i metodu řešení, podrobnější bloková schémata jsou k dispozici u řešitelů, nejsou však součástí tohoto materiálu. V následujících odstavcích popíšeme postupně úpravu mikroprogramu Z, mikroprogramy nových operací PČ, další možnosti úspory místa v kvadrantu II a konečně zrychlené algoritmy devíti nejčastěji používaných instrukcí EC 1021.

3.1. Úprava mikroprogramu Z

Nejrozsáhlejším a nejnáročnějším úkolem je úprava mikroprogramu Z, který zaujímá asi polovinu kvadrantu II a účastní se provádění všech instrukcí počítače. Cílem této úpravy je:

a) zařazení dalších 14 přípustných operačních znaků formátu RX
b) zrychlení základního cyklu dekódování instrukce
c) úspora místa v kvadrantu II.

Podrobnější rozbor ukázal, že zařazení dalších přípustných operačních znaků do dosavadního algoritmu Z je sice možné, ale obtížné a hlavně nevede k žádným dalším úsporám místa ani času. Na možnost změny algoritmu Z poukázal už před časem ing.V.Tichý z VÚMS, svůj návrh však podrobněji nerozpracoval. Použili jsme některých jeho základních myšlenek, rozpracovali jsme je však zcela samostatně, některé z nich jsme později museli opustit jako neproveditelné a naopak jsme nalezli jiná, účinnější řešení.

Původní algoritmus Z jsme rozdělili na 6 logicky uzavřených úseků:

1. dekódování operačního znaku s kontrolami,

2. výpočet efektivních adres,
3. vytváření požadavků na přerušení při chybách,
4. obsluhu timeru a přenos požadavků na vnější přerušení,
5. zjišťování požadavků na přerušení v registru IPR, a konečně
6. vlastní přerušení, spojené s obsluhou tlačítek.

Každý z těchto úseků byl zkoumán samostatně a nakonec přeprogramován. Postupně tak vzniklo několik verzí, které stále lépe splňovaly tři základní požadavky. V současné verzi mají úseky 1., 2. a 5. zcela nové algoritmy, úseky 3., 4., 6. byly přeprogramovány jen s ohledem na úsporu místa, protože probíhají jen vzácně (méně než 1 procento provedených instrukcí) a nemají tedy podstatný vliv na rychlost počítače. Zato v hlavní větvi úseků 1,, 2., 5. a zčásti i 6. stála na prvním místě dynamická úspora času, protože po této cestě probíhají všechny instrukce vůbec.

Nový algoritmus úseku 1. je podstatně jednodušší než původní, vyžaduje však více místa v paměti ROS, protože místo tří rozskokových tabulek po 16 pozicích obsahuje jedinou tabulku dlouhou 128 adres. Vcelku zabírá asi o 30 adres více, část těchto adres je však možno využít ještě k jiným účelům (zhruba 20 adres) a hlavně je asi o 7 usec rychlejší. Po přečtení operačního znaku z paměti a jednoduché úpravě se provádí rozskok na 128 různých pozic. Všechny přípustné operační znaky leží totiž v polovině řádků úplné tabulky 256 možných kombinací. Úprava OZ je tedy provedena tak, aby se do téže pozice rozskokové tabulky promítl vždy jeden OZ z "přípustného" a jeden z "nepřípustného" řádku tabulky. Rozlišení těchto dvou operačních znaků se provádí až na konci úseku 2. současně s ostatními kontrolami jediným (a tstem. 49 ze 128 pozic rozskokové tabulky

odpovídá v každém případě nepřípustných OZ; protože při chybě OZ není třeba šetřit časem (výpočet bude stejně přerušen a vzápětí ukončen) mohou být v těchto částech rozskokové tabulky umístěny souvislé části jiných mikroprogramů a teprve ve vhodném místě se testem rozliší, zda na tato místa mikroprogram přišel rozskokem anebo odjinud.

Na těch pozicích, které odpovídají platným operačním znakům, se do pracovní buňky v zápisníku připraví adresa první mikroinstrukce příslušného mikroprogramu, kam se odskočí na konci 2. úseku. Do sčítačky se připraví jednoduše testovatelná kombinace, která udává formát instrukce a také, je-li třeba přičítat obsah indexregistru k efektivní adrese. Tato kombinace se pak také uloží do zápisníku. Další pokračování závisí na typu instrukce. Instrukce DIG a IDL projdou (spolu s ostatními privilegovanými) ještě testem na přípustnost a pak odskakují přímo na příslušné mikroprogramy. U instrukcí BC a BCR se podle masky a CC rozhodne, zda se skok uplatní nebo ne a podle toho připraví adresa první mikroinstrukce mikroprogramu plnění PC anebo se PC zvýší o délku instrukce a pokračuje přímo na závěr mikroprogramu Z (úsek 5.). U instrukcí pohyblivé čárky se přímo vytvoří "operační znak" a odešle do JPČ, která může operaci připravit. Všechny instrukce mimo IDL, DIG a neúčinných skoků pak pokračují do úseku 2.

Úsek 2. obsahuje běžný výpočet efektivní adresy (adres), která se ukládá stejně jako dosud do pracovní oblasti v zápsiníku. Rozskoky podle formátů (RR, RS a SI, RX s nenulovým indexem a SS) se dějí podle formátové slabiky, připravené v úseku 1. Nakonec se naplní nový obsah do PC, zkontroluje alignement a bit X'20 '(resp.X'10') operačního znaku, který byl při prvním rozskoku ignorován (nepřípustná polovina tabulky op.znaků) a

provede rozskok na první mikroinstrukce příslušných specifických mikroprogramů. Chyby, zjištěné během dekódování instrukce (op.znak, privilegovaná instrukce apod.) se uplatní tak, že po zjištění chyby se změní rozskoková adresa mikroprogramu, který pak z úseku 2. nepokračuje na mikroprogram jednotlivé instrukce, nýbrž do příslušného bodu úseku 3. (nahazování chybových bitů v IPR). Tím je zaručeno, že i po chybě má PC správný obsah a odpadá poměrně velká "záplata" původního mikroprogramu Z. Další značná úspora místa vzniká tím, že instrukce se společným počátkem vlastního mikroprogramu mohou mít společnou i jeho první mikroinstrukci; tak např. všechny operace PČ vedou do čtyř (místo 14) počátečních bodů. Časová úspora v úseku 2. není velká, je však možno jednoduše zabudovat mikroprogramovou ochranu paměti supervizoru, softwarově kompatibilní s ochranou čistě hardwarovou, obsaženou ve verzi 6.

Úseky 3. a 4. zůstávají bez hlubších změn, byly staticky zkráceny a části mikroprogramů budou umístěny v oblasti nepřípustných OZ rozskokové tabulky úseku 1. Při přeprogramování se ušetřila řada adres v jiných kvadrantech ROS, které nelze nyní využít, tvoří však rezervu pro případné opravy chyb.

K závažným změnám došlo v úseku 5. - prohlídka registru IPR a zjištění požadavku na přerušení. V původním algoritmu se v hlavní větvi prohlížely všechny čtyři byty IPR s testem na nulovost. Při nenule se přecházelo na podrobnější prohlídku, při níž se uplatnila také maska z IMR. Nový algoritmus je rovněž dvouúrovňový, je však v hlavní větvi téměř o 3 usec rychlejší a asi o 30 adres kratší. Je založen na novém využití bitu 23. v IPR, který slouží pro čistě interní vteřinovou značku timeru. Jakmile se tento signál přečte z bloku T (v úseku 4.), odečte

se jednička od čitače času a bit se do IPR vůbec nepřenáší. V novém algoritmu je 23. bit IPR využit jako logický součet všech 32 bitů IPR; tento logický součet se vytváří mikroprogramem a zkráceně označuje SUMIPR. Hlavní větec úseku 5. tedy tvoří pouze test na nulovost bitu SUMIPR. Je-li tento bit nenulový, větví se mikroprogramem na vedlejší větev, kde se SUMIPR nejprve znuluje, aby sám nevedl na přerušení. Pak se cyklicky probírají jednotlivé byty IPR, podle jejich obsahu se připravuje nový obsah SUMIPR, maskují se IMR a při nenulovém součinu se vypočte váha přerušení, bit s nejvyšší prioritou se vymaže a přechází se do úseku 6. Byly-li všechny byty IPR nulové, zůstane nulový i SUMIPR a následující instrukce budou probíhat pouze zkrácenou hlavní větví.

Bit SUMIPR se nahazuje

- při mikroprogramovém zjištění chyby (úsek 3.)
- při přenosu z registru T do IPR (nenulový signál TC)
- pokaždé, když se uplatnilo přerušení (úsek 6.).

Prodlouženou větví se tedy prochází zpravidla dvakrát na každý požadavek přerušení v IPR, z toho podruhé zpravidla zbytečně. Prodlouženou větví se však prochází vždy, je-li přerušení zamaskováno. Proto má jistý význam i to, že zdržení, způsobené prohlídkou IPR v těchto případech je také proti původnímu mikroprogramu značně menší a tvoří max. 7 usec. Úsek 6. zůstává téměř beze změn.

3.2. Mikroprogramy PČ

Vlastní mikroprogramy PČ jsou rozděleny do dvou částí; první z nich je tvořena příslušnými pozicemi rozskokové tabulky úseku 1 a navazujícím mikroprogramem vytvoření "operačního znaku" JPČ, druhou tvoří dva mikroprogramy - mikroprogram operací

uložení registru (STE a STD) a mikroprogram "vysílacích" operací. Oba mikroprogramy jsou značně podobné. Tvoří je vyklus pro příjem (vysílání) operandu po bytech mezi hlavní pamětí a blokem W resp. JPČ. Parametrem cyklu je délka operandu v hlavní paměti (4 nebo 8 byte), krátký operand se při vysílání doplní čtyřmi nulovými byte (mantissa je v přímém kódu), při přijímání se poslední čtyři byte ignorují. Mikroprogram vysílacích operací nakonec ještě vyčká na příchod podmínkového byte z JPČ. Tento byte obsahuje podmínkový kód výsledku u operací slučování a porovnání, případně příznak přetečení exponentu resp. dělení nulou. U operací násobení a dělení by bylo možno odložit převzetí tohoto příznaku až do příští instrukce PČ. Tím by se dalo ušetřit poměrně mnoho času - mikroprogramová část instrukce PČ trvá cca 25 - 30 nsec - je však třeba ještě uvážit další důsledky tohoto vnitřního sdílení času. Konečně rozhodnutí souvisí mj. s obvodovým návrhem interface a signálů.

Další dosud nedořešenou otázkou je zabezpečení přenosu mezi ZJ a JPČ, neboť blok W není vybaven paritou. Rozhodnutí padne mezi třemi možnými alternativami:

- blok W doplnit o paritu
- paralelně k bloku W připojit nový blok s paritou
- přenos zabezpečit mikroprogramově kontrolním součtem nebo podélnou osmibitovou paritou.

Řešení bude ovlivněno také vnitřním zabezpečením dat uvnitř JPČ.

3.3. Další možnosti rozšíření

V průběhu práce na mikroprogramech kvadrantu II se ukázalo, že i v jiných částech tohoto kvadrantu lze poměrně snadno ušetřit několik desítek adres. Protože mikroprogram Z nebude patrně delší a mikroprogramy PČ se vejdou z větší části do ušetřeného místa mikroprogramu Z, zvážili jsme ještě možnost úpravy mikroprogramů některých nejčastěji používaných a přitom jednoduchých operací. Oba problémy spolu tedy souvisejí: při úspoře místa bude možno podstatně zrychlit další operace.

Místo lze ušetřit v mikroprogramu IPL a mikroprogramu operací prc kanály, které zaujímají skoro celý zbytek kvádru II. Vedle nich jsou zde už jen kratičké mikroprogramy operací IDL a MVI, na nichž nelze mnoho ušetřit. Při úpravách zmíněných mikroprogramů je ovšem třeba zvýšené opatrnosti, neboť právě chování kanálů je značně záludné a není nikde popsáno. Proto jsme postupovali jinou metodou. Nerozebírali jsme logiku celých mikroprogramů, nýbrž vytypovali celkem jedno místo v mikroprogramech kanálových operací a přeprogramovali jen tyto úseky s požadavkem naprosté totožnosti vstupního i výstupního "interface" těchto krátkých úseků. Přes tento velmi přísný požadavek se nám podařilo ušetřit více než 40 adres. Nové úseky těchto mikroprogramů se přitom s hlediska svého okolí chovají stejně jako původní, takže riziko chyby je téměř vyloučeno.

Souběžně s touto činností jsme podle dostupných statistických údajů vybrali dalších sedm často používaných operací a napsali pro ně nové mikroprogramy. Po instrukci BC, o jejímž novém algoritmu jsme už hovořili, se nejčastěji používá instrukce MVC. Ukázalo se, že její mikroprogram lze zrychlit ze 6.7 usec/byte na 4.4 usec/byte tím, že se zvýší překrývání práce hlavní paměti. Na dalších místech co do frekvence použití jsou

jednak velmi krátké operace typu LA, TM, CLI, MVI apod., které se podstatně zrychlí zrychlením základního cyklu a v jejichž mikroprogramech nelze mnoho ušetřit. Jednak se tu vyskytují velmi významné operace L, LH, ST, STH, které jsou v dosavadních mikroprogramech celkem nešťastně skloubeny do jediného mikroprogramu s dalšími instrukcemi (LSP, SSP), jejichž význam pro rychlost stroje je nepatrný. Navrhli jsme proto dva nové mikroprogramy (LH, L, LM a STH, ST, STM), které spolu se zkrácením základního cyklu zrychlí tyto operace na dvojnásobek.

Další možnost, o níž jsme uvažovali, je mikroprogramová ochrana paměti supervizoru (resp. SPOOL - oblasti). Přestože hardwarová ochrana paměti ve verzi 6 ROSu je dokonalejší, výrobní závod ji odmítá dodatečně montovat do dosud vyrobených kusů. Proto má i mikroprogramové řešení svůj smysl pro VZLÚ. Mikroprogramová ochrana paměti by chránila začátek paměti až do adresy, zakódované v masce ochrany, proti přepsání s výjimkou přepsání od kanálů a přepsání prvních max.255 byte instrukcí, které přechází přes maximální kapacitu na začátek (wrap-around). Kódování masky ochrany bude v supervizoru MOS vyřešeno tak, aby obojí způsob ochrany (obvodová i mikroprogramová) vedl ke stejným výsledkům a byl tedy "kompatibilní". Mikroprogramová ochrana paměti prodlouží trvání všech instrukcí, které zapisují do paměti, jednorázově o cca 2,4 usec.

Vzhledem k tomu, že software počítače 1C21 je dnes hotov a ve VZLÚ se programuje převážně ve vyšších jazycích, jsme neuvažovali o dalším rozšiřování operačního kódu směrem k úplnému kódu JSEP. Přidání několika instrukcí nemá v dané situaci smysl a úplný operační kód by si vyžádal rozšíření ROS a velké množství práce.

Poznámka: v tabulce 3 uvádíme předpokládané časy některých

operací po navržené úpravě. Časy jsou vypočteny na základě pesimistického odhadu trvání mikroinstrukce 400nsec. Staré časy jsou časy naměřené programem s přesností, danou přesností timeru.

4. HLAVNÍ PLÁNOVACÍ ÚDAJE

V této kapitole jsou uvedeny hlavní plánovací údaje, jakož i stručné ekonomické zhodnocení celého záměru.

4.1. Harmonogram prací

Je uveden na obr.2. První a druhá větev činnosti odpovídají dvěma hlavním částem celého úkolu, totiž problematice optimalizace mikroprogramů a problematice vlastní výroby, odlaďování a zavedení JPČ. Ve spodní části harmonogramu jsou uvedeny požadované termíny dodávek materiálu popř. stavebních prvků, které budou nakoupeny.

Práce uvedené v harmonogramu bude zajišťovat kolektiv deseti pracovníků v poloviční pracovním úvazku. To při průměrném počtu 22 pracovních dní v měsíci a předpokládané průběžné době 22 měsíců znamená 22.22.10.4,25 = 20.600 hodin technických pracovníků. Při plánované sazbě 50,83 Kčs za jednu T-hodinu to znamená zhruba 1,05 mil.Kčs nákladů. Pro úplnost dodejme, že z toho činí čisté mzdové fondy 285.000 Kčs.

4.2. Materiál a kooperace

Přehled materiálu potřebného k realizaci je uveden v tabulce 4 a to v první části materiál resp. stavební prvky tuzemské, v druhé části integrované obvody zahraniční výroby.

Náklady na práce v kooperaci se odhadují částkou Kčs 150.000 a zahrnují zejména výrobu desek a simulátoru, jakož i kabeláže.

Další nákladovou položkou této kategorie je 50 hodin strojového času počítače Siemens 4004,který je zapotřebí k odlaďování a simulaci mikroprogramů. Při ceně 4.500 Kčs za strojovou hodinu činí tato nákladová položka celkem 225.000 Kčs.

Rovněž je třeba započítat ztrátu strojového času počítače EC 1021 při instalaci JPČ. Při 5 dnech s 15-hodinovým čistým provozem a sazbou 1500 Kčs za strojovou hodinu to činí 5.15.1500 = 112.500 Kčs.

4.3. Rekapitulace nákladů a zhodnocení ekonomické efektivnosti

Celkovou rekapitulací jednotlivých nákladových položek uvedených v předchozích odstavcích resp. tabulkách dospíváme k tomuto přehledu (zaokrouhleno na tisíce Kčs)

1. Hodiny technických pracovníků (z toho mzdový fond Kčs 285.000,-)	**1,050.000,- Kčs**
2. Materiál a kupované stavební prvky (z toho dovoz z KS Kčs 26.670)	**141.200,- Kčs**
3. Kooperace (VÚMS,Metra,ZPA)	**150.000,- Kčs**
4. Strojový čas počítače Siemens 4004	**225.000,- Kčs**
5. Ztráta stroj.času EC 1021 při instalaci JPČ	**113.000,- Kčs**
Celkem	**1,680.000,- Kčs**

Při odhadu ekonomické efektivnosti realizace JPČ vycházíme z těchto předpokladů:

a) Předpokládáme, že čistými výpočty v pohyblivé řádové čárce bude počítač vytěžován v průměru pouze max.0,5 hod. denně. Při očekávaném 50-násobném zrychlení výpočtů ve srovnání s dosavadním programovým způsobem to znamená, že za tuto dobu se

vytvoří hodnota výpočtů odpovídající době provozu 0,5 x 50 = = 25 hodin. Při očekávané fakturační ceně 1500 Kčs za strojovou hodinu počítače EC 1021 dochází tak k dennímu přírůstku produkce v hodnotě (25 - 0,5) . 1500 = 36750 Kčs. To znamená návratnost pořizovacích nákladů za $\frac{1,680.000}{36.750}$ = 45,7 pracovních dní = cca 2 měsíce. K tomuto způsobu ekonomického hodnocení lze namítnout, že je sice formálně správný, avšak trvalý provoz počítače EC 1021 v režimu pohyblivé čárky realizované programově by byl zřejmě nehospodárný i pro provozovatele tohoto počítače. Proto učiníme porovnání ještě druhým způsobem, a to

b) porovnáním výkonu soustavy EC 1021 + JPČ s počítačem podobných výkonových parametrů.

Uvažujeme opět, že fakturační cena jedné strojové hodiny počítače EC 1021 činí 1500 Kčs. Soustava EC 1021 + JPČ bude mít v uvažované konfiguraci výkonové parametry srovnatelné s počítači, jejichž fakturační cena strojové hodiny je 4500 Kčs. Předpokládáme-li, že úlohy s instrukcemi v pohyblivé řádové čárce budeme v průměru počítat 4 hodiny denně, (tj. čisté operace v pohyblivé čárce k ostatním jsou v poměru 1:7), bude denní přírůstek produkce dán částkou 4.(4500-1500) = 12 000 Kčs. To znamená návratnost pořizovacích nákladů za $\frac{1,680.000}{12.000}$ = 140 pracovních dní = cca 6 měsíců.

Vezmeme-li méně příznivý výsledek z obou způsobů porovnání, docházíme k závěru, že plánované doplnění počítače EC 1021 jednotkou pohyblivé čárky je z ekonomického hlediska velmi efektivní a vynaložené náklady se vrátí v relativně krátké době 6 měsíců rutinního provozu.

5. ZÁVĚREČNÉ ZHODNOCENÍ

Návrh na doplnění počítače EC 1021 o aritmetiku v pohyblivé řádové čárce, podrobně popsaný v této zprávě, lze stručně charakterizovat těmito zásadními údaji:

1. Vlastní aritmetika pohyblivé čárky (dále JPČ) bude vytvořena jako čistě obvodový celek s vlastním řadičem a zdroji. Bude obsahovat také čtyři rychlé 8-bytové registry PČ a bude vestavěna ve dvou standardních buňkách do skříně ZJ 1021. Napájecí zdroj bude rovněž v této skříni.

2. Jednotka pohyblivé čárky bude převážně realizována na integrovaných obvodech TESLA s použitím malého počtu obvodů střední integrace z dovozu. Přednost mají typy, které se budcu vyrábět i u nás. Pro zjednodušení bude JPČ pracovat pouze s operandy dvojné přesnosti (délka 8 byte). Čas slučování nejvýše 50 usec, násobení max. 200 usec. K základní jednotce 1021 bude JPČ informačně (logicky) připojena přes upravený blok externího styku (blok W), který bude doplněn o paritní bit. Styk bude paralelní na 8+1 bitů, tj. 1 byte s paritou.

3. Mikroprogramy počítače 1021 budou upraveny přešitím celého kvádru II (celkem 512 slov) paměti ROS. Úpravou algoritmů mikroprogramu základní jednotky (mikroprogram Z) se ušetří místo pro zabudování dalších 14 instrukcí PČ: plnění a uložení registru, sčítání, odčítání, násobení, dělení a porovnání. Tyto instrukce budou realizovány pouze ve formátu RX (registr - paměť), varianta RR nebude realizována hardwarově. Operace jednoduché přesnosti budou mikroprogramem převedeny na dvojné (doplněním operandu o nulové byty mantisy).

4. Vedlejším produktem úpravy mikroprogramů bude zrychlení ostatních operací počítače, u kratších operací (B,LA,MVI apod.)

až o 25 %. V ostatních 5 kvádrech ROS dojde nanejvýš k nepatrným změnám, které bude možno provést přešitím na místě.

5. Vlastní instalace je rozvržena tak, že úplné vysazení počítače nebude trvat déle než 1 týden. Protože formát dat zůstane zachován, bude možno JPČ ihned využívat se stávajícím software EC 1021 s nepatrnými změnami, které budou rovněž provedeny v rámci realizace JPČ. Jde o změnu rozvoje makroinstrukcí PČ a změnu interpretu pro programy přeložené z FORTRANu.

6. Z rozboru očekávaných nákladů a přínosů vyplývá návratnost pořizovacích nákladů v relativně krátké době 6 měsíců.

DODATEK

Seznam členů řešitelského kolektivu

Jan S o k o l (VÚMS Praha) - celkový návrh řešení, vedení práce na mikroprogramech, ověřování výsledků po instalaci

Ing. Pavel F a n t a (VÚMS) - návrh a simulace mikroprogramů, výroba a ověřování hardware

Jan H o u d e k (ZPA Košíře) - detailní návrh schématu JPČ, výroba a oživování

Ing. Jiří K o u s a l (VÚMS) - návrh a kontrola mikroprogramů, testování a instalace

Ing.Vl. N a v r á t i l (VÚMS) - návrh a kontrola mikroprogramů, ověřování a zkušení provoz

Jiří P e l o u c h (VÚMS) - programy pro simulaci a zkoušení, výrobní podklady a kontrolu, úpravy software

Vojtěch S e d l á č e k (ZPA Čakovice) - celkový návrh řešení, vedení práce na obvodové části JPČ, účast na výrobě a ověřování

Ing. Otakar Š t a s t n ý (ZPA Košíře_-obvodový návrh aritmetiky, radiče a interface, výroba a oživení

Václav T r o j a n (VÚMS) - návrh a simulace mikroprogramů, úpravy software, účast při instalaci

Ing. Vladimír V r b s k ý (ZPA Čakovice) - obvodový návrh zdrojové a jisticí části, připojení k ZJ, výroba, oživení a vlastní instalace.

Práce na dokumentaci se účastní všichni členové řešitelského kolektivu.

ZÁZNAM O OPONENTNÍM ŘÍZENÍ

Oponentní řízení ke zprávě se konalo dne 15.5.1975 ve VZLÚ za přítomnosti oponentů, jimiž byli
Ing. Libor Obruča, Výzkumný ústav matematických strojů Praha,
Ing. Zbyněk Beneš, ZPA Čakovice

Závěrečné zhodnocení:

1. Technické řešení jednotky pro realizaci operací v pohyblivé čárce i zvolená alternativa řešení jsou z hlediska požadovaných výkonových parametrů i z hlediska konkrétní realizace hodnoceny kladně. Za značný přínos ve využití počítače EC 1021 byly označeny úpravy standardních mikroinstrukcí, které vedou ke zrychlení operačních rychlostí počítače, a to bez závislosti na používání dotyčné doplňkové jednotky. Dalším výrazným zlepšením proti současnému vybavení počítače EC 1021 je ochrana supervizoru. Rovněž byl oceněn způsob připojení jednotky k počítači, který vyžaduje minimální nárok na odstavení počítače z provozu při uvádění jednotky do provozu.

2. Bylo konstatováno, že postup prací pokračuje podle harmonogramu, ve většině případů s mírným předstihem. Pro úplnost byly do harmonogramu zahrnuty i práce, které zajišťuje VZLÚ.

3. Připomínky oponentů se týkaly převážně obecnějšího využití jednotky a nemají pro realizaci ve VZLÚ zásadní význam. Navrhované řešení je vhodné pro realizaci jednoho až dvou kusů pro VZLÚ, jak bylo zadáno při zahájení prací na úkole.

4. Řešitelé doplní návrh jednotky pohyblivé čárky tak, aby mohly být přiměřeným způsobem zvládnuty situace uvedené v posudku Ing.Beneše ad d), tj. nulování jednotky v případě chyby a předčasného ukončení programu z řadiče ZJ počítače.

5. Nároky na realizaci pokud jde o pracovní kapacity, materiálové, výrobní a kooperační položky jsou přiměřené. Celkový ekonomický přínos vyplývající z realizace jednotky pohyblivé čárky lze očekávat velmi výrazný.

Ing. Oldřich K r o p á č, CSc
vedoucí odboru 3500

Tabulka 1.

POROVNÁNÍ DOBY ŘEŠENÍ MATICOVÝCH ÚLOH NA POČÍTAČÍCH MINSK 22 a EC 1021

Program pro násobení matic A.B=C. Jednotlivé případy se liší řádem matice, typem proměnných a počtem nulových prvků

řád	typ	příklad		sestavení		výpočet	
		MINSK	1021	MINSK	1021	MINSK	1021
40	INTEGER	0:43	1:17	1:43	0:45	3:21	4:56
40	REAL	0:45	1:18	1:46	0:45	3:32	10:27
40	REAL	0:47	1:14	1:46	0:46	3:42	10:51
20	DOUBLE	0;48	0:43	2:27	0:30	3:39	4:31
20	DOUBLE	0:50	0:44	2:28	0:33	3:46	6:09

Časy měřeny stopkami v minutách:vteřinách.

Tabulka 2.

SEZNAM OPERACÍ POHYBLIVÉ ČÁRKY

přesnost		funkce
jednoduchá	dvojitá	
LE	LD	plnění registru
STE	STD	uložení registru
AE	AD	přičtení k registru s norm.
SE	SD	odečtení od registru s norm.
CE	CD	normalizované porovnání
ME	MD	násobení
DE	DD	dělení

Podmínkový kód mění operace přičtení, odečtení a porovnání. K přetečení exponentu může dojít při operacích přičtení, odečtení, násobení a dělení. Přetečení exponentu stejně jako dělení nulou vede k přerušení výpočtu.

Tabulka 3.

PRŮMĚRNÉ ČASY NĚKTERÝCH OPERACÍ V MIKROSEKUNDÁCH

operace	nový čas cca	starý čas
LE	32	510
LD	37	600
AE, SE, CE	37	2 000
AD, SD, CD	42	2 400
MD	200	7 000
DD	200	40 000
zrychlení stávajících operací:		
BCR bez skoku	10	13
se skokem	11	18.5
BC bez skoku	11	28
se skokem	18	30
LH	20	40
L	23	48
STH	19	40
ST	22	49
MVC	19+4.4/byte	28+6.7/byte

ostatní operace:	zrychlení
formát RR	cca o 3 μsec
RS,SI,RX bez indexu	cca o 11 μsec
RX s indexem	cca o 12 μsec
SS	cca o 10 μsec

Operace s pohyblivou čárkou ve váženém průměru, tj. 0.55 L/ST, 0.35 A/S/C, 0.06 M a 0.04 D, se zrychlí 60krát /50 usec proti 3050 usec programové PČ.

Tabulka 4.

PŘEHLED POŽADOVANÉHO MATERIÁLU

a) materiál domácí produkce

pol.	název - označení	množství	č.výkresu	výrobce	cena za 1 ks	cena celkem
1	(zdroj) stabilizátor ABP 102 8,5V/10A	1 ks		ZPA Děčín	6.350,-	6.350,-
2	stabilizátor AUP 101 5V/4,8A	3 ks		ZPA Děčín	2.810,-	8.430,-
3	konektor KO 48 A	35 ks		ZPA Trutnov	42,-	1.470,-
4	konektor KO 48 B	35 ks		ZPA Trutnov	27,-	945,-
5	paměťový blok R 512 (obsah dle našich podkladů)	1 ks	G3-12801-2	Metra Blansko	37.800,-	37.800,-
6	buňka levá	2 ks	G3-11879-2	ZPA Čakovice	250,-	500,-
7	matiční deska	2 ks	G3-26170-2	Strojírny Nový Bor	1.000,-	2.000,-
8	vodítko dl. 130 mm	70 ks	materiálové číslo 403 920	Výr.družstvo STYL Praha-Prosek	0,53	37,-
9	kuprextit dvoustranný tl.1,5mm	2 m^2 (12 kg)			85,-/1 kg	1.020,-
10	číslic.integr.obvody (specifikace typů bude upřesněna později)	700 ks		Tesla Rožnov	80,-	56.000,-
				Materiál domácí c e l k e m		114.552,-

TABULKA „INFORMAČNÍ"

Tabulka 4. - pokračování

b) integrované obvody zahraniční výroby (Texas Instruments Inc)

pol.	označení	název	množství	cena DM za kus x)	celkem	cena Kčs celkem xx)
11	SN7483AN	4-bit binary full adders	2	12,20	24,40	201,30
12	SN74H87N	4-bit true/complement,zero/ one element	2	10,80	21,60	178,20
13	SN7491AN	8-bit shift register	8	10,40	83,20	686,40
14	SN7489N	64-bit read/write memory	4	54,80	219,20	1.808,40
15	SN74126N	gates with tri-state totem pole outputs	11	6,90	75,90	626,20
16	SN74193N	synchronous 4-bit up/down counter	8	19,20	153,60	1.267,20
17	SN74194N	4-bit bidirestional univ. shift register	16	15,00	240,00	1.980,00
18	SN74S157N	data selector multiplexers	1	41,50	41,50	342,80
19	SN74S134N	gates with 3-state totem pole outputs	2	15,60	31,20	257,40
		Materiál zahraniční celkem			890,60	7.347,90
		Materiál c e l k e m				121.900,00 Kčs

x) Podle ceníku z r.1972 včetně 30% přirážky čs.obchodním organizacím

xx) Přepočteno podle kursu 1 DM = 8,25 Kčs

Tabulka 4 - pokračování

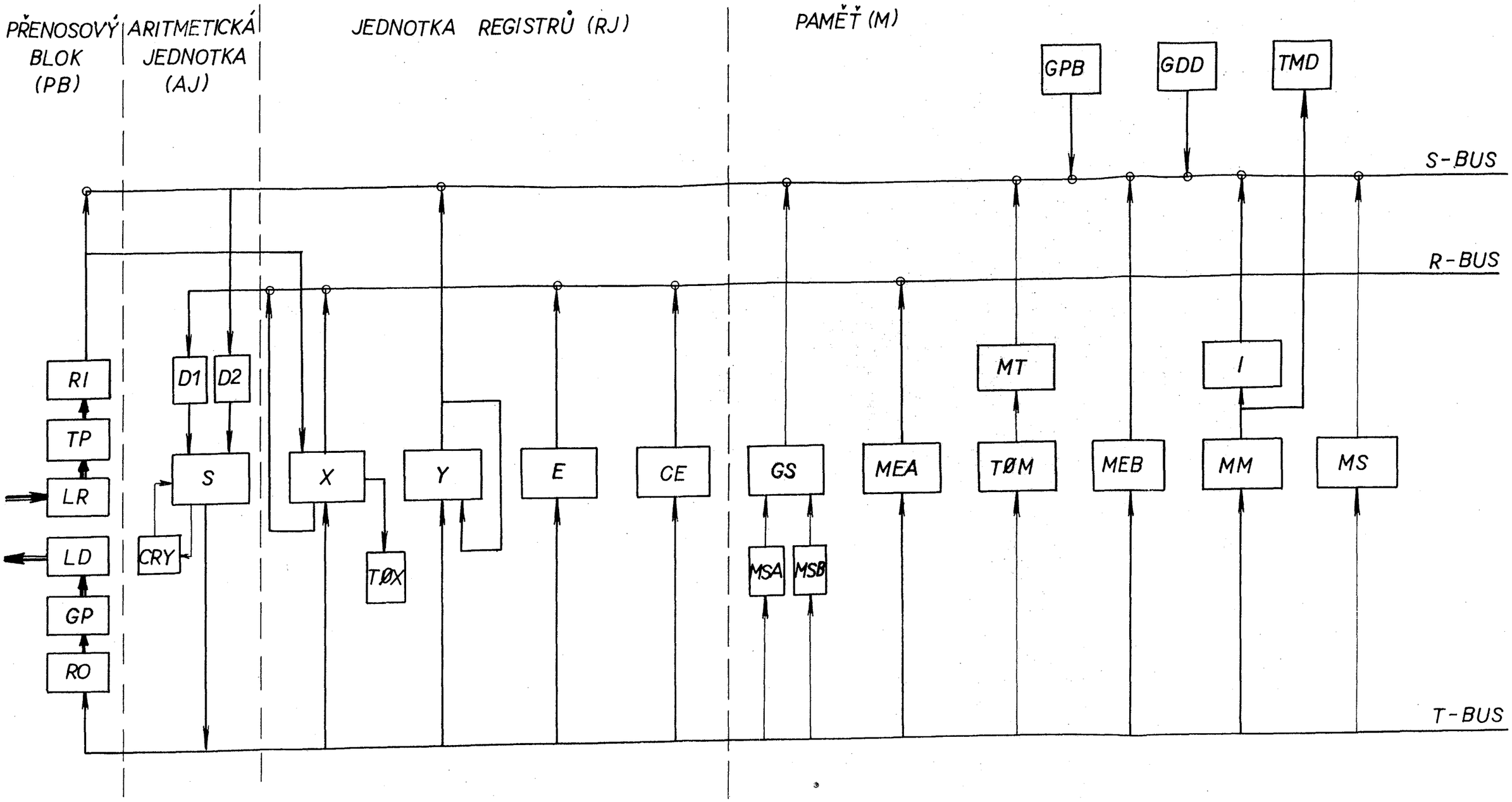
b) integrované obvody zahraniční výroby (texas Instruments Inc) - doplněný seznam

pol.	označení a název		množství	cena Kčs	x) celkem
11	SN7442AN	4-LINE-TO-10-LINE DECODER			1.440,-
12	SN7483AN	4-BIT BINARY FULL ADDER			200,-
13	SN7485N	4-BIT MAGNITUDE COMPARATOR			2.040,-
14	SN74H87N	4-BIT TRUE/COMPLEMENT, ZERO/C[illegible]			360,-
15	SN7489N	64-BIT READ/WRITE MEMORY			2.500,-
16	SN74126N	GATE WITH 3-STATE TOTEM-POLE		-	700,-
17	SN74154N	4-LINE-TO-16-LINE DECODER/DEM[illegible]	14	-	2.380,-
18	SN74S157N	QUADRUPLE 2-LINE-TO-1-LINE DAT[illegible]R/MULTIPLEXER	6	-	900,-
19	SN74164N	8-BIT PARALLEL-OUT SERIAL SHIFT REGISTER	16	160,-	2.560,-
20	SN74165N	PARALLEL-LOAD 8-BIT SHIFT REGISTER	1	200,-	200,-
21	SN74180N	9-BIT ODD/EVEN PARITY GENERATOR/CHECKER	2	200,-	400,-
22	SN74193N	SYNCHRONOUS 4-BIT UP/DOWN COUNTER	14	150,-	2.100,-
23	SN74194N	4-BIT BIDIRECTIONAL UNIVERSAL SHIFT REGISTER	32	320,-	10.240,-
24	SN74LS266N	QUADRUPLE 2-INPUT EXCLUSIV/NOR GATE WITH OPEN-COLLECTOR OUTPUTS	10	65,-	650,-

x) Podle cen[illegible] z[illegible] četně
30% přir[illegible]
orgar[illegible]
podle kursu 1

Materiál zahraniční celkem 26.670,-

Materiál c e l k e m 141.222,-



LEGENDA:

SPOJENÍ PŘES 3 STATE LOGIC

ŠÍŘKA TOKU 1 BIT

" " 4 BITY

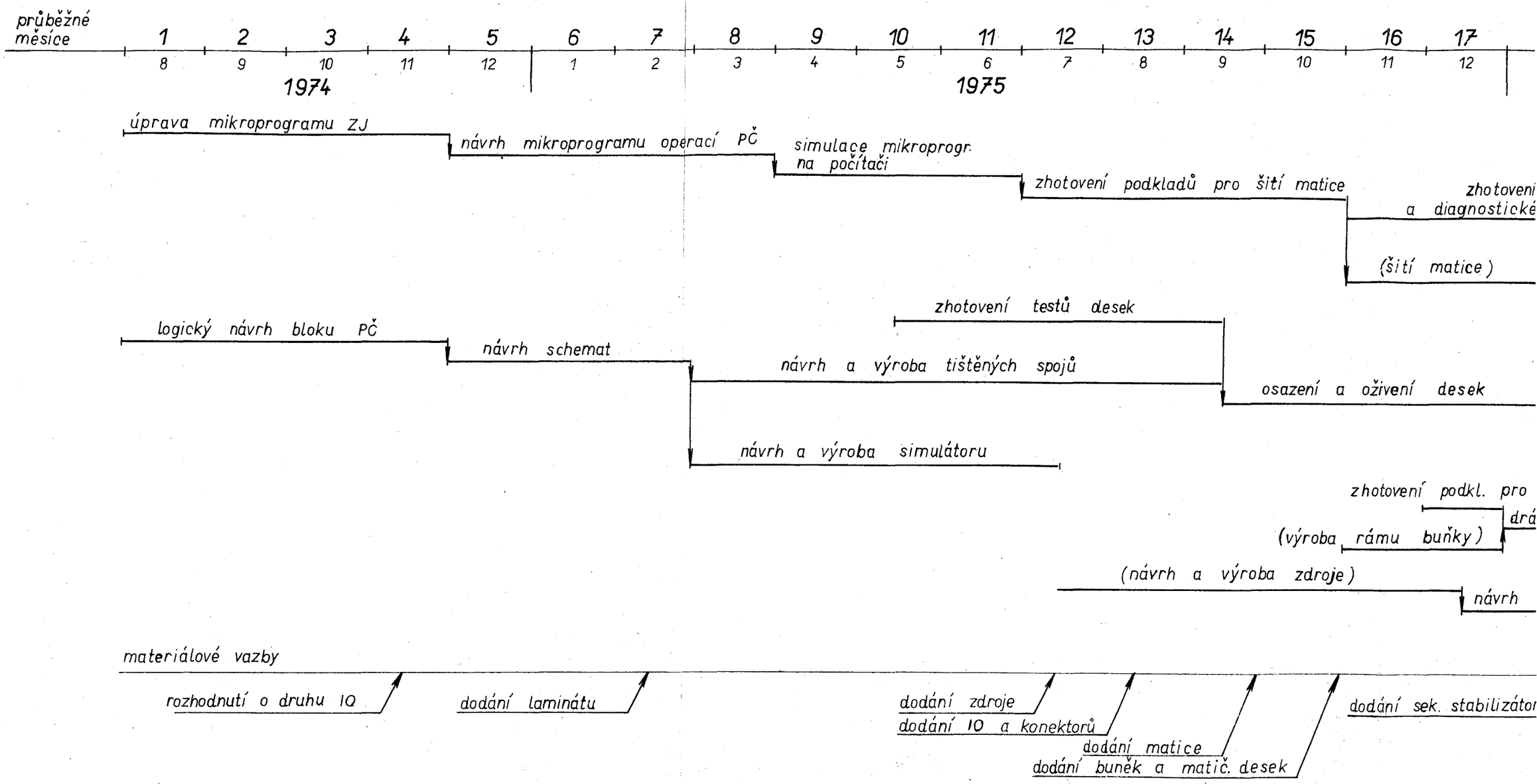
" " 8 BITŮ

" " 9 BITŮ

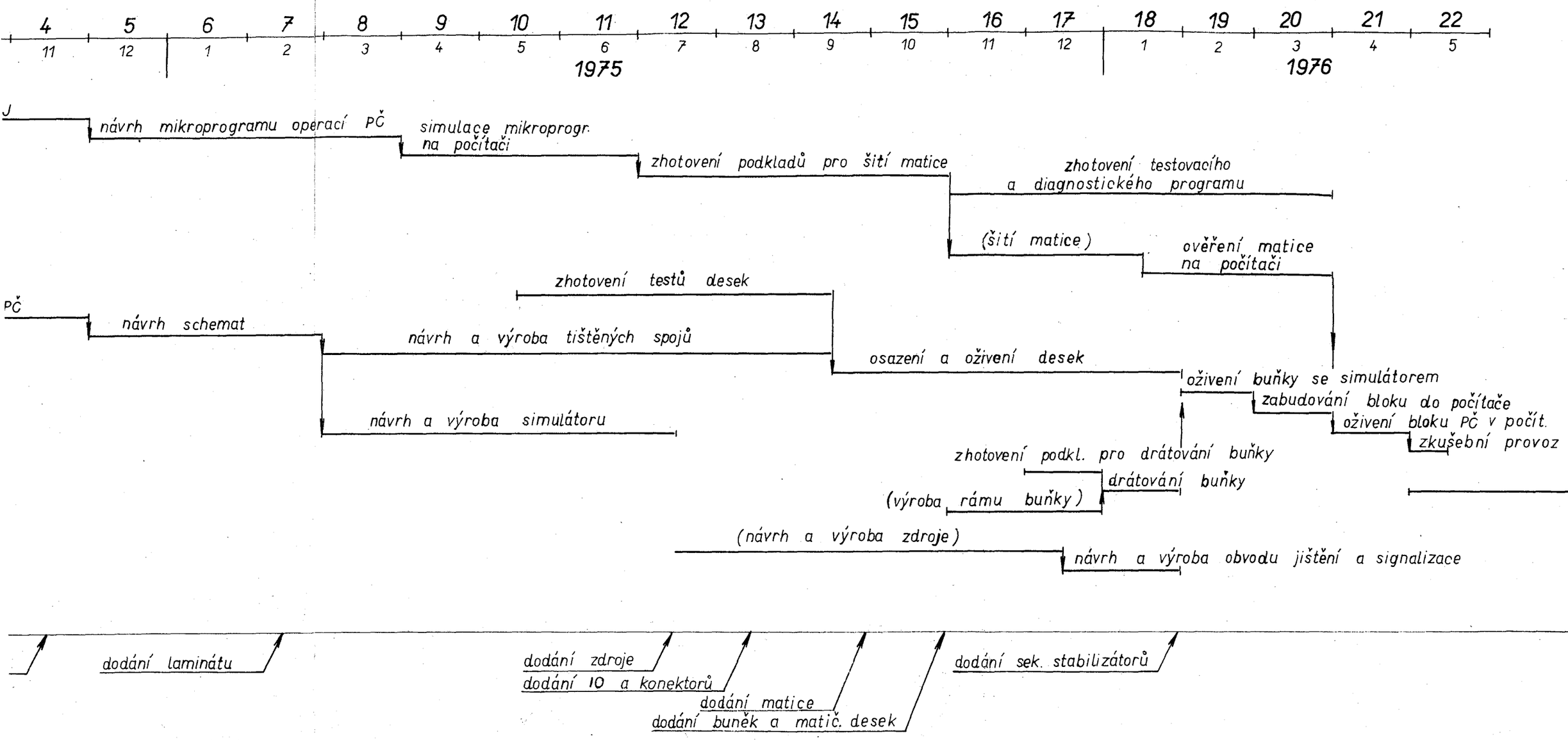
OBR. 1. BLOKOVÉ SCHÉMA JPČ

Legenda k blokovému schemanu JPČ (obr.1)

AJ - aritmetická jednotka
CE - reverzibilní čítač charakteristiky
CRY - obvod pro přenos v aritmetické jednotce
D1,D2 - obvody pro tvorbu doplňků
E - operační registr charakteristiky
GDD - generátor podílu
GP - generátor parity
GPB - generátor podmínkového byte
GS - generátor znaménka
I - invertor
LD - výstupní obvod
LR - vstupní obvod
M - paměť
MEA - paměť charakteristiky operandu A
MEB - paměť charakteristik operandu B
MM - paměť mantis operandu B
MS - paměti znamének operandu B
MT - paměti testu nulovosti mantis
MSA - paměť znaménka mantisy operandu A
MSB - paměť znaménka operandu B
PB - přenosový blok
RI - registr vstupní
RJ - jednotka registrů
RO - výstupní registr
S - sčítačka
TMD - testovací obvod násobitele
TØM - test nulovosti mantisy
TP - test parity
TØX14 - testovací obvod nulovosti nejvyšší čtveřice bitů registru X
X - operační registr obousměrný
Y - operační registr jednosměrný



OBR. 2. ČASOVÝ ROZVRH REALIZACE JPČ



OBR. 2. ČASOVÝ ROZVRH REALIZACE JPČ

HARMONOGRAM PRACÍ PŘI NÁVRHU A VÝROBĚ DVOU KUSŮ JEDNOTKY POHYBLIVÉ ČÁRKY

(dodatek z 20.5.1975)

1. dílčí etapa: 31.7.1975
   - návrh a výroba simulátoru pro statické oživení JPČ

2. dílčí etapa: 30.9.1975
   - návrh podkladů k výrobě tištěných spojů
   - dohotovení a ověření desek s tištěnými spoji - 1.kus
   - zhotovení testů jednotlivých desek

3. dílčí etapa: 31.12.1975
   - osazení a oživení desek aritmetiky - 1.kus
   - dohotovení a ověření desek s tištěnými spoji - 2.kus
   - zhotovení podkladů pro zapojení buňky

4. dílčí etapa: 31.3.1976
   - osazení a oživení desek řadiče - 1.kus
   - zapojení buňky a oživení pomocí simulátoru - 1.kus
   - zhotovení podkladů pro úpravu zdrojové soustavy

5. dílčí etapa: 30.6.1976
   - oživení a zkušební provoz JPČ v počítači - 1.kus
   - osazení a oživení desek aritmetiky - 2.kus

6. dílčí etapa: 30.9.1976
   - osazení a oživení desek řadiče - 2.kus
   - zapojení buňky a oživení pomocí simulátoru - 2.kus

7. dílčí etapa: 30.11.1976
   - oživení a zkušební provoz JPČ v počítači - 2.kus
   - zhotovení úplné dokumentace JPČ

Oponentský posudek na
výzkumnou zprávu VZLÚ

Ing.Libor Obruča
VÚMS-Vokovice

Sokol,Sedláček: NÁVRH NA DOPLNĚNÍ HARDWARE POČÍTAČE EC-1021 OPERACEMI V POHYBLIVÉ ČÁRCE

Posuzovaná výzkumná zpráva má charakter ideového návrhu technické realizace dodatečného zabudování obvodů pro zpracování operandů v pohyblivé čárce (dále jen PČ), do základní jednotky (ZJ) počítače EC-1021.

Ve zprávě je uveden stručný přehled alternativních možností řešení. Vybraná varianta řešení respektuje všechny omezující podmínky realizace, kterými jsou :

- pokud možno minimální zásahy a změny do existujícího obvodového řešení ZJ EC-1021
- termín realizace v polovině roku 1976
- možnosti realizace, tj. přiměřené náklady na materiál a práci a skutečnost, že řešitelé a realizátoři jsou vázáni svojí hlavní pracovní náplní k jiným organizacím.

Z tohoto hlediska považuji navrhované řešení za téměř optimální. Zejména myšlenka připojení adaptéru- nazývaného ve zprávě "jednotka pohyblivé čárky" (JPČ)- k ZJ prostřednictvím bloku W, pro přímé propojení EC-1021 s jiným systémem, je nápaditá. Významným přínosem je rovněž přepracování mikroprogramů standardních instrukcí počítače EC-1021, obzvláště pak spojovacího mikroprogramu Z. Tímto zásahem, vynuceným potřebou získat kapacitu v paměti mikroinstrukcí, se získá pozoruhodné zvýšení operačních rychlostí počítače. Podle odhadů ve zprávě je zvýšení cca o 25% a rychlost zpracování supervizorových programů o cca 80 %. Samozřejmě je evidentní zvýšení rychlosti operací s PČ (tj. cíl řešení), které ve srovnání s programovanou PČ je o jeden a půl řádu vyšší. Rovněž je přínosem odzkoušení mikroprogramové ochrany rezidentních částí supervizoru v operační paměti.

Až potud tedy naprostý souhlas.

Připomínky mám k :

1. - volbě výběru některých typů integrovaných obvodů(IO)
2. - neúplnosti souboru instrukcí PČ
3. - prověřování funkční správnosti JPČ v provozu
4. - úpravě W bloku
5. - některým položkám rekapitulace nákladů(kap.4.3) a harmonogramu prací

Ad 1.- Zvolené typy IO jsou zřejmě pro konkretní jednorázové řešení JPČ ve VZLÚ vybrány optimálně. Toto lze z předloženého ideového návrhu pouze předpokládat, protože kromě blokového schema JPČ není konkretnější řešení předvedeno. Z požadavku na dovoz vyplývá, že se pro řešení chtějí použít mj. typy SN74H87N, SN74126N, SN74194N, SN74S134N, SN74157N . Tyto typy nejsou ani v perspektivních výrobních programech závodů LDS, natož TESLY ROŽNOV. Je pravda, že jde sice o relativně velmi malá množství, ale mají charakter tzv. trvale dovozních součástek z devizové oblasti. Pro jednorázové řešení nebo nanejvýše jen u několika zájemců to asi nehraje roli. Za určitých okolností tato skutečnost však může způsobit nepříjemné komplikace. To je důvod proč se domnívám, že by bylo prozíravější zvážit možnost řešení s prvky , které jsou ve výběru pro tzv. JSEP I a II. Popř.dokázat těžkopádnost takového řešení anebo jeho nemožnost . Každopádně by měly být trvale devizové součástky minimalizovány.

Ad 2.- Navržené omezení souboru instrukcí PČ na 2x 7 instrukcí, je provedeno zřejmě co do typů instrukcí optimálně. Počet instrukcí je evidentně omezen nedostatkem volné kapacity v řídící paměti (RP) a asi také pracovní kapacitou. Vzhledem k tomu, že se řeší ve VÚMS překonstruování RP z transformátorové na polovodičovou , by tím byla k dispozici až dvakrát větší kapacita této paměti i větší rychlost paměti. Paměť bude na prvcích ROM. V takovém případě by při zachování navrhované koncepce, tj. JPČ adaptované k ZJ prostřednictvím W bloku, přímého propojení- úplný operační kód značně zvýšil hodnotu (bohužel i cenu) a přitažlivost této úpravy pro celou řadu budoucích i existujících uživa-

telů EC-1021. Je ovšem jasné, že toto doplnění na úplný operační kód by si vyžádalo nejen doplnění instrukcí pohyblivé čárky, ale i některých instrukcí binární aritmetiky - tedy poměrně rozsáhlá práce, která je mimo pracovní možnosti řešitelů.

Ad3.- Podle kap. 2.5 je obvodové zajištění JPČ provedeno inverzní logikou a příčnou paritou. Doporučuji, aby signál "dobroty" stavu JPČ - HOTOVOJPČ byl testován v L bloku, tak jak je alternativně navrhováno. Zásah do řídící smyčky je choulostivější a asi by se neměl použít.
V této kapitole a ani jinde ve zprávě není zmínka o systému prověřování správné funkce obvodů JPČ. Např. krátkým mikrodiagnostickým testem, který by bylo možno iniciovat pomocí instrukce DIG. Tyto instrukce by byly např. součástí zvláštního testovacího programu, prověřujícího JPČ. V mikrodiagnostickém programu by se měly prověřit především pomocí režimu ZR správné funkce "hlídačů" parity.
Toto profylaktické a diagnostické vybavení by nemělo v návrhu chybět.

Ad 4.-Pokud jde o připravovanou úpravu W bloku, chci upozornit na to, že nelze podcenit potřebu zachování původní funkce tohoto bloku.Tvrzení na straně 8. zprávy, že "...ztráta možnosti propojení více počítačů nebyla pro VZLÚ kritická...", je zřejmě pro VZLÚ platná. Při širším uplatnění těchto úprav v počítačích EC-1021 by to však mohlo způsobovat komplikace. Desky W a KWP by se měly překonstruovat takovým způsobem , aby byla funkce W bloku zachována bez nutnosti výměny desek. V aplikacích při propojení počítačů by mohlo být užitečné mít k disposici možnost využívat PČ současně.
Obvyklou námitkou pro používání možností W bloku pro svoji původní funkci bývá, že není k disposici potřebný supervizor a operační systém. Realizace zde diskutovaného řešení adaptace systému pro PČ a úpravy dohlížecích programů jsou však důkazem toho, že je možné leccos a tedy i využívání původní funkce W bloku není nereálné.
Z tohoto pohledu by možná zabezpečení přenosu mezi JPČ a ZJ realizovaná pomocí mikroprogramového kontrolního

součtu udělalo nejméně komplikací. Zřejmě je tento způsob ale pomalý. Rovněž možnost ochrany podélnou paritou by se měla zvážit.
Při této příležitosti bych se zmínil o formální nejednotnosti informací poskytovaných zprávou o využití W bloku. Informace na str.8,12,25 a 33 nejsou vzájemně sladěny.

Ad 5.-V kapitole 4.3"Rekapitulace nákladů a zhodnocení ekonomické efektivnosti" chybí podle mého názoru náklady na práci spojenou s výrobou desek, provedením kabeláže a výrobou simulátoru (o kterém je zmínka jen v harmonogramu). Tyto práje nutné vykázat "papírově", protože budou kooperovány. Sestava pracovníků řešících úlohu by tyto práce nebyla schopna provést aniž by se dopustila přestupků proti právně ekonomickým předpisům.
V harmonogramu prací chybí činnosti týkající se úpravy MOS.

Závěrem konstatuji, že ideový projekt řešení návrhu technické realizace adaptace obvodové PČ do ZJ EC-1021, pro jednorázové realizování ve VZLÚ plně vyhovuje. Pokud by se předpokládalo širší uplatnění pak je podle mého názoru nutné věnovat pozornost připomínkám a podle toho projekt doplnit.

V Praze, 27.3.1975

Ing. Libor Obruča

Libor Obruča

Oponentský posudek na
výzkumnou zprávu VZLÚ

Ing. Zbyněk Beneš
ZPA - Čakovice

Sokol, Sedláček: NÁVRH NA DOPLNĚNÍ HARDWARE POČÍTAČE EC-1021 OPERACEMI V POHYBLIVÉ ČÁRCE

Výzkumná zpráva se zabývá ideovým návrhem technické realizace dodatečného zabudování hardwerových prostředků pro zpracování operandů v pohyblivé řádové čárce /dále PČ/ do základní jednotky /ZJ/ počítače EC-1021.

Ve zprávě jsou stručně naznačena možná alternativní řešení. Vybrané řešení si všímá všech omezujících podmínek. Jsou to:
a/ minimální zásahy do stávajícího obvodového řešení ZJ EC-1021,
b/ termín uvedení do provozu nejdéle do poloviny roku 1976,
c/ možnost realizace jak po nákladové, tak po pracovní stránce.

Vzhledem k těmto podmínkám je zvolené řešení podle mého názoru optimální. Řešení adaptéru, nazývaného ve zprávě "jednotka pohyblivé čárky" /JPČ/, jako samostatného celku, který pracuje v jistém slova smyslu ve sdílení času, je na místě. Zkrátí se tím značně časy jednotlivých instrukcí PČ. Rovněž připojení adaptéru /JPČ/ pomocí W bloku pro přímé spojení EC-1021 s jinými systémy, je vhodné. Velkým přínosem je přepracování dosud používaných mikroprogramů počítače EC-1021, hlavně pak spojovacího mikroprogramu Z. Tento zásah je vynucen potřebou získat místo v paměti mikroinstrukcí a vede k zvýšení rychlosti počítače. Ve zprávě jsou uváděny hodnoty zkrácení doby výpočtu o 25% jako celku a při zpracování supervisorových programů až o cca 80%. Je třeba si rovněž povšimnout mikroprogramové ochrany rezidentních částí supervisoru v operační paměti, které řešení nabízí.

V těchto bodech naprosto souhlasím s řešiteli.

Připomínky lze uplatnit:

a/ k chování upraveného mikroprogramu Z v případě absence JPČ,
b/ ke kontrole funkční správnosti JPČ,
c/ k průběhu vlastního spojení JPČ a ZJ,
d/ k hlášení chyb a předávání podmínkového kódu.

Ad a/ Vzhledem k tomu, že úprava II. kvádru řídící matice, která vede k nezanedbatelnému zrychlení počítače EC-1021, bude lákavá i pro uživatele, pro které nemá vlastní JPČ význam, vyvstává otázka, jak se bude chovat toto uspořádání. Zpráva se tímto problémem nezabývá zřejmě z nedostatku času.

Ad b/ Kontrola vlastní funkce JPČ je provedena inversní logikou a příčnou paritou. Informace o stavu bloku JPČ je předávána při každém styku s JPČ. Chyba JPČ tak nevede k zastavení stroje, jako u ostatních bloků ZJ, ale pouze k přerušení programu. V tom případě je nezbytná optická signalizace stavu JPČ, která by byla vyvedena na panel technika. Bližší informace pak poskytuje signalizace v samotném bloku JPČ.
Ve zprávě však není zmínka o možnostech systémového prověřování správné funkce JPČ. Jistě by bylo možné udělat krátký program na test JPČ, který by pomocí instrukce DIG vyvolal krátké mikrodiagnostické testy na prověření obvodů JPČ. Pomocí režimu ZR by se dali prověřit hlídací obvody parity. Vlastní návrh by se měl touto otázkou zabývat.

Ad c/ Spojení JPČ s ZJ probíhá přes W blok. Využívá se přitom signálů MOW a WM3. Bližší popis tohoto spojení však chybí.
Při návrhu této části je třeba mít na zřeteli vlastnosti počítače EC-1021, který je řešen tak, že přenosová jednotka /PJ/ počítače využívá bloky ZJ. Práce PJ probíhá tak, že PJ přerušuje v případě potřeby ZJ. Přerušení však nemůže nastat v době, kdy ZJ pracuje s operační pamětí. Vzhledem k tomu, že pro spojení JPČ a ZJ jsou využity signály, které ovládají operační paměť, nemohla by v té době PJ přerušit toto spojení. V případě, že by signály byly voleny v rychlém sledu za sebou, mohlo by dojít k přeběhům při spolupráci s vnějšími pamětmi. Dále není ve zprávě zmínka o tom, jak se projeví přerušení od PJ v době přenosu informací mezi JPČ a ZJ.

Ad d/ Blok JPČ pracuje ve sdílení času. To znamená, že informace o dokončení operace je k dispozici až při dalším volání JPČ. Tento způsob vede ke zkrácení doby potřebné pro jednotlivé instrukce. Programátoři však musí brát tuto okolnost v úvahu a počítat s ní při sestavování programů.
Může však dojít k situaci, kdy dojde k chybě v bloku JPČ. Než však je možno vybrat toto hlášení, je program z jiných důvodů předčasně ukončen. V tom případě se toto hlášení projeví až v následujícím programu, který bude volat JPČ a dojde k jeho přerušení. V takovýchto případech by měla existovat možnost manuelního nulování JPČ od panelu operátora. Programový způsob ošetření těchto situací by si vyžádal zásah do supervizoru. Tam by se mohla zabudovat část, která by před každým voláním programu nulovala JPČ v případě, že by byl přítomen. Skutečnost, že JPČ není přítomen, by se u ostatních počítačů nikterak neprojevila, aby se nemusely udržovat dvě odlišné verse supervizoru.

Na závěr shrnuji, že ideový projekt řešení návrhu na doplnění hardware počítače EC-1021 operacemi PČ pro ústav VZLÚ plně vyhovuje. V případě, že by přicházelo v úvahu širší uplatnění, je podle mého názoru třeba brát na zřetel výše zmíněné připomínky.

V Praze, 12. 5. 1975

Ing. Zbyněk Beneš

Zbyněk Beneš