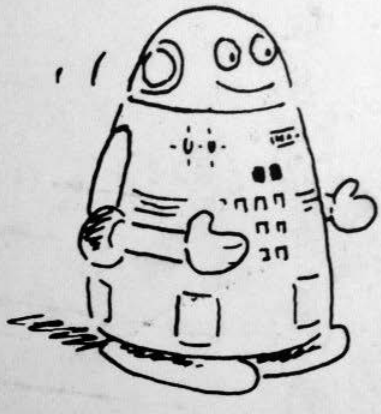

ZÁKLADNÍ PROGRAMOVÉ VYBAVENÍ PRO MIKROPOČÍTAČ

ONDRA

-Uživatelská příručka-

Koncepce programového vybavení

V mikropočítači ONDRA je obsažena pevná paměť ROM, která je určena pro rezidentní program. Na obsah této paměti jsou z hlediska programátora kladeny různé protichůdné požadavky. Na jedné straně existuje spousta užitečných podprogramů, které by bylo vhodné mít v paměti ROM, naproti tomu je kapacita této paměti podstatně menší, než by bylo potřeba - 4 KB. Je tedy nutné vybrat pouze nejdůležitější podprogramy. Které to jsou? Na to má každý programátor svůj vyhraněný názor. Naše koncepce vychází z těchto hledisek:

- je třeba vybrat procedury, které by se opakovaly ve všech systémových i uživatelských programech (což určitě není výpis obsahu paměti v šestnáctkové soustavě a podobné oblíbené funkce)
- ONDRA je určen především dětem, tedy laickým uživatelům a nikoli programátorům
- ONDRA je český mikropočítač, musí tedy umět psát česká a slovenská písmena s diakritickými znaménky a to malé i velké abecedy
- do paměti ROM se nevejde žádný systémový program (BASIC, PASCAL...), na začátku práce je tedy nutné nahrát program z magnetofonu
- magnetofon je nespolehlivé záznamové zařízení, je nutné mít možnost opakovat čtení při chybě
- některé používané magnetofony nemají počítadlo otáček, je tedy nutná snadná orientace na kazetě
- při psaní na klávesnici často dochází k překlepům, je proto třeba mít možnost je snadno opravit - to nejlépe umožňuje obrazovkový editor
- ONDRA je vhodný do mikropočítačových učeben, kde bude více mikropočítačů propojeno do sítě - je nutné umožnit vstup programů a dat ze seriové linky (z jiného ONDRY nebo centrálního mikropočítače)

Do paměti ROM jsme tedy umístili tyto podprogramy:

- vstup z klávesnice
- výstup na obrazovku
- čtení z magnetofonu
- zápis na magnetofon
- obrazovkový editor
- vstup ze seriové linky
- výstup do seriové linky
- zavaděč programů ve strojovém kódu

Jak psát na klávesnici

Při prvním pohledu na klávesnici zjistíme, že tlačítek je podstatně méně než je obvyklé. Tím je dáno, že každé tlačítko bude mít několik funkcí. Funkce se volí pomocí speciálních kláves - přeřaďovačů. Je jich pět a mají následující označení a funkci:

(SHIFT)	přeřaďovač pro psaní velkých písmen (bez něho se píší malá písmena)
(ALT)	přeřaďovač pro psaní znaků (popis na klávesách nahoře vlevo)
(ČS)	přeřaďovač pro psaní písmen s diakritickým znaménkem
(Ø-9)	přeřaďovač pro psaní číslic (popis nahoře vpravo) a dalších znaků, které na klávesách nejsou nakresleny
(CTRL)	přeřaďovač pro psaní speciálních řídících kódů

Postup volby funkce pomocí přeřaďovače je následující:

- stiskneme a držíme přeřaďovač
- stiskneme významové tlačítko
- pustíme současně obě tlačítka (nebo dříve významové tlačítko, přeřaďovač můžeme držet déle)

Přeřaďovač pro psaní písmen s diakritickým znaménkem (ČS) má poněkud odlišný způsob použití - po stisknutí ho můžeme pustit a pak teprve stisknout významové tlačítko. To je důležité pro psaní velkých písmen s diakritickým znaménkem. Postup je pak následující:

- stiskneme a pustíme ČS
- stiskneme a podržíme SHIFT
- stiskneme písmeno
- pustíme písmeno a přeřaďovač

Výsledkem této operace je velké písmeno s diakritickým znaménkem.

Po nechtěném stisknutí ČS můžeme jeho funkci zrušit opětovným stiskem ČS. Sudý počet stisknutí tedy nedělá nic, při lichém počtu stisknutí diakritické znaménko platí.

Někdy je vhodné psát většinou velkými písmeny a jen občas malými. Kombinací kláves CTRL a SHIFT dojde k výměně velkých a malých písmen. Bez SHIFTu se nyní píší velká písmena, se SHIFTem malá. Opětovným stiskem kombinace se vrátí původní přiřazení.

Postup:

- stiskneme a podržíme CTRL
- stiskneme SHIFT
- pustíme obě tlačítka (v libovolném pořadí)

Při stisknutí klávesy nebo přeřaďovače se ozve pípnutí, které signalizuje správné stisknutí klávesy.

Po stisku klávesy se vypíše jeden znak. Jestliže klávesu podržíme déle, po chvíli se začne tento znak opakovat. Tato funkce (nazývá se autorepeat) je vhodná například pro podtrhávání nebo, a to hlavně, pro pohyb kurzoru po obrazovce (viz. kapitola Obrazovkový editor).

Tabulka významů tlačítek s různými přeřaďovači

klávesa	(nic)	SHIFT	ALT	Ø-9	ČS	SH-ČS	CTRL
A	a 61	A 41	- 2D	~ 7E	á C1	Á E1	Ø1
B	b 62	B 42	? 3F	? 3F	C2	E2	Ø2
C	c 63	C 43	: 3A	7F	č C3	Č E3	Ø3
D	d 64	D 44	= 3D	6Ø	ď C4	Ď E4	Ø4
E	e 65	E 45	# 23	3 33	é C5	É E5	Ø5
F	f 66	F 46	^ 5E	| 7C	r C6	R E6	Ø6
G	g 67	G 47	_ 5F	_ 5F	C7	E7	Ø7
H	h 68	H 48	< 3C	< 3C	u C8	U E8	Ø8
I	i 69	I 49	(28	8 38	í C9	Í E9	Ø9
J	j 6A	J 4A	> 3E	> 3E	ů CA	Ů EA	ØA
K	k 6B	K 4B	[5B	{ 7B	l CB	L EB	ØB
L	l 6C	L 4C	] 5D	) 7D	l CC	L EC	ØC
M	m 6D	M 4D	. 2E	. 2E	o CD	O ED	ØD
N	n 6E	N 4E	, 2C	. 2C	ň CE	Ň EE	ØE
O	o 6F	O 4F	) 29	9 39	ó CF	Ó EF	ØF
P	p 7Ø	P 5Ø	@ 4Ø	Ø 3Ø	o DØ	O FØ	1Ø
Q	q 71	Q 51	! 21	1 31	a D1	A F1	11
R	r 72	R 53	$ 24	4 34	ř D2	Ř F2	12
S	s 73	S 54	+ 2B	+ 2B	š D3	Š F3	13
T	t 74	T 54	% 25	5 35	ť D4	Ť F4	14
U	u 75	U 55	' 27	7 37	ú D5	Ú F5	15
V	v 76	V 56	; 3B	; 3B	D6	F6	16
W	w 77	W 57	" 22	2 32	é D7	É F7	17
X	x 78	X 58	/ 2F	\ 5C	a D8	A F8	18
Y	y 79	Y 59	& 26	6 36	ý D9	Ý F9	19
Z	z 7A	Z 5A	* 2A	* 2A	ž DA	Ž FA	1A
nový_řádek	ØD	1B	ØD	ØD	AD	BB	ØD
nahoru	8Ø	84	88	8C	2Ø	24	1C
vlevo	81	85	89	8D	21	25	1D
vpravo	82	86	8A	8E	22	26	1E
dolů	83	87	8B	8F	23	27	1F

Na mikropočítači ONDRA je mimo klávesnici ještě jedno tlačítko. Je umístěno ze strany na levé bočnici. K

jeho stisknutí je potřeba použít špičatý nástroj, například tužku. Tomuto tlačítku se říká RESET. Tímto tlačítkem lze kdykoliv přerušit běžící program nebo proces čtení z kazetového magnetofonu. Je to takzvaný teplý start (WARM START) programu - program v paměti zůstane zachován i se všemi svými daty.

Pokud před stiskem tohoto tlačítka stiskneme a podržíme klávesu se šipkou vlevo, vpravo nebo dolů, výsledný efekt této operace bude následující:

šipka vpravo	- studený start programu
šipka vlevo	- restart programu
šipka dolů	- studený start systému

Studený start programu je jeho počáteční nastavení. Všechny data nebo program (např. v jazyce BASIC) jsou vymazány - zapomenuty.

Studený start systému je jeho počáteční inicializace - jako po zapnutí. Obsah paměti se přitom samozřejmě nemaže, takže program, který byl v paměti, zůstává zachován a lze ho opět spustit pomocí restartu programu.

Restart programu je jeho opětovné spuštění po předchozím studeném startu systému. Program se v tomto případě spouští teplým startem.

Stejný význam jako tlačítko RESET má i současné stisknutí tlačítek D, F a G.

Obrazovkový editor

V rezidentntí paměti ROM je obsažen i jednoduchý obrazovkový editor. Ten nám umožňuje psaní a základní opravy textu.

Místo na obrazovce, na které bude vypsán následující znak, je označeno blikajícím obdélníčkem, kterému se říká kurzor.

Písmena, symboly, číslice se po stisku příslušné klávesy napíší na pozici kurzoru a ten se posune doprava na následující pozici. Znak pod kurzorem a všechny znaky vpravo od kurzoru, pokud tam nějaké jsou, se posunou doprava - napsaný znak se vloží de textu na místo kurzoru. Tomuto módu se říká automatické vkládání.

Kurzorem je možno po obrazovce pohybovat pomocí kláves s šipkami následujícím způsobem:

šipka vlevo	o znak doleva
šipka vpravo	o znak doprava
šipka nahoru	na předchozí řádek
šipka dolů	na následující řádek
SHIFT vlevo	na začátek řádky
SHIFT vpravo	na konec řádky
SHIFT nahoru	do levého horního rohu obrazovky

Špatně napsaný znak je možné vymazat:

CTRL nahoru	výmaz znaku pod kurzorem
CTRL vlevo	výmaz znaku vlevo od kurzoru
CTRL vpravo	výmaz do konce řádky
CTRL dolů	výmaz do konce obrazovky

Při pohybování kurzorem po obrazovce pomocí šipek nebo při mazání většího počtu znaků velice oceníte opakování znaků - autorepeat.

Čtení programu z kazety

Jedním ze zařízení, ze kterých umí ONDRA po zapnutí číst programy, je kazetový magnetofon. Každý soubor (program nebo obecná data) na kazetě má svůj název, podle kterého ho můžeme najít a identifikovat. Pokud chceme načíst nějaký konkrétní program, napíšeme jeho název. Můžeme také napsat

pouze část názvu (začátek). Pokud nenapíšeme žádný název, bude se číst první program, který ONDRA na kazetě najde.

Jak tedy načteme program z kazety ?

Po zapnutí ONDRY se vyvolá obrazovkový editor, pomocí kterého napíšeme název programu, který chceme načíst. Vlastní čtení se spustí po stisknutí klávesy NOVÝ_ŘÁDEK (tlačítko se zalomenou šipkou dolů a vlevo - vpravo druhá shora). Jako název se vezme text, který je napsán na řádce s kurzorem.

Na následující řádce se vypíše název (pokud byl uveden), několik mezer, typ souboru- ".KÓD" (viz. dále) a číslo požadovaného bloku - "Ø1". Řádek bude vypadat například takto:

PASCAL .KÓD Ø1

Současně se zapne relé povolující otáčení magnetofonu a rozsvítí se žlutá LED dioda vlevo od klávesnice. Obrazovka začne rychle blikat - to indikuje, že ONDRA chce číst z magnetofonu.

Vložíme do magnetofonu kazetu a pustíme přehrávání. Pokud je kazeta přetočena přesně před začátkem zvoleného programu, po nalezení tohoto začátku obrazovka zhasne a bude se číst první blok programu. Jestliže na kazetě v daném místě je jiný program nebo prostředek programu, vpravo od názvu čteného programu (popsáno výše) se vypíše název, typ a číslo bloku souboru, který je nahrán v tomto místě kazety. Tak můžeme průběžně sledovat, ve které části kazety jsme. Pro rychlejší nalezení požadovaného programu můžeme použít rychlého převíjení vpřed nebo vzad.

Po nalezení začátku požadovaného programu zhasne obrazovka na pár sekund, po které se čte blok programu. Potom se obrazovka rozsvítí a vypíše se celý název čteného programu a číslo dalšího bloku (Ø1, Ø2, Ø3...). Po jeho nalezení obrazovka opět zhasne a čte se další blok. Toto se opakuje až do načtení celého programu.

Po načtení celého programu se vypne relé, takže se zastaví magnetofon, a zhasne žlutá LED dioda. Načtený program se automaticky spustí, vypíše svoji úvodní hlavičku a je připraven k použití.

Pokud máme připravenu kazetu na správném místě, celá operace načtení programu spočívá pouze ve stisknutí klávesy NOVÝ_ŘÁDEK a puštění magnetofonu! Tato jednoduchost a účelnost ostře kontrastuje se systémem nahrávání programů na jiných československých mikropočítačích.

Magnetofon je nespolehlivé zařízení, zvláště pokud se jedná o lacinější typ nebo pokud používáme nekvalitní kazety. Proto při čtení může dojít k chybě. Tato chyba je signalizována textem "Chyba při čtení", který se vypíše do pravé části řádky a pípnutím. V tomto případě samozřejmě nemusíme opakovat celé čtení (jak je to pohříchu obvyklé na jiných mikropočítačích), ale stačí opakovat čtení bloku, ve kterém došlo k chybě. Přetočíme kazetu o malý kousek zpět a zkusíme číst blok znovu.

Vyzkoušeli jsme u mikropočítače ONDRA různé typy kazetových magnetofonů (K10, M710, Geracord GC 6020, Unitra MK 232 P, Tesla SM 260) a výskyt chyby při čtení byl velice řídký. Problém může způsobit pouze čtení na jiném magnetofonu s podstatně odlišně nastavenou kolmostí hlavy nebo při velkém kolísání rychlosti vlivem deformované hnací kladky.

Typ souboru

Každý soubor nahraný na kazetě má své jméno a typ. Jméno nám umožňuje rozlišit různé soubory, typ určuje příslušnost souboru k nějakému systémovému nebo uživatelskému programu. Typ souboru si určuje každý program sám, není třeba ho při zápisu ani čtení uvádět, i když je to možné. Pokud chceme uvést speciální typ, napíšeme ho za název a oddělíme ho tečkou. Například takto:

NÁZEV.TYP

Různým systémovým a uživatelským programům tedy přísluší různé typy souborů. V současnosti jsou definovány tyto typy souborů:

typ	program	obsah souboru
KÓD	EDTASM	zaveditelný program ve strojovém kódu (pro mikroprocesor Z-8Ø nebo 8Ø8Ø)
ASM	EDTASM	zdrojový text pro editor-assembler
BAS	BASIC	zdrojový text pro interpret jazyka Basic
KAR	KAREL	zdrojový text pro program Karel
LSP	LISP	zdrojový text pro interpret jazyka Lisp
PAS	PASCAL	zdrojový text pro překladač jazyka Pascal
PLG	PROLOG	zdrojový text interpretru jazyka Prolog

Přerušení čtení programu

Někdy je třeba v průběhu čtení programu (nebo už při hledání začátku) přerušit proces čtení. To je možné stisknutím tlačítka RESET. Tím je čtení přerušeno a opět máte k dispozici obrazovkový editor pro zápis jiného názvu nebo jiné akce.

Tlačítko RESET lze použít kdykoliv a přerušení činnosti se provede ihned. Pro stejný účel je možné i současné stisknutí tlačítek D, F a G. Pokud se právě čte blok dat (zhasnutá obrazovka), čtení se dokončí a až po rozsvícení obrazovky se projeví stejný účinek jako u tlačítka RESET.

Výstup na obrazovku

Další část rezidentní paměti ROM zabírají podprogramy pro psaní na obrazovku. Psát na obrazovku potřebuje každý program. Proto jsme se snažili poskytnout programátorům i uživatelům co největší možnosti, pro usnadnění a urychlení

jejich práce. Základní charakterisktika se dá shrnout do těchto bodů:

- zobrazení 21 řádků po 40 znacích
- znaky v matici 12x8 bodů
- písmena velké i malé abecedy i s českými a slovenskými diakritickými znaménky (kód KOI-8 čs 2)
- normální/inverzní zobrazení (bílé na černém nebo naopak)
- speciální funkce:
  - adresace kurzoru na souřadnice X a Y
  - pohyb kurzoru do všech směrů
  - přesun kurzoru na začátek/konec logické řádky
  - výmaz/vsunutí znaku do řádky
  - výmaz/vsunutí řádky
  - výmaz do konce řádky/obrazovky

Konkrétní řídící kódy budou popsány dále.

Okna

Výstup na obrazovku může být přepínán do různých oken. Co to okna vlastně jsou ?

Okno je myšlený obdélník na obrazovce, do kterého se píše. Okno je definováno umístěním (souřadnice levého horního rohu) a rozměry (počet znaků v řádce a počet řádek). Po zapnutí mikropočítače ONDRA jsou všechna okna (je jich celkem osm) nastavena na přes celý displej, což jsou následující pozice a rozměry:

- levý horní roh je na začátku první řádky
- počet znaků na řádce je maximální, tedy 40 znaků
- okna jsou do konce obrazovky, to je 21 řádek
- toto nastavení lze číselně označit: (0,0, 40,21)

Každé z osmi oken je možné nastavit na jinou pozici a jiné rozměry. Neprovádí se žádná kontrola překrývání oken - pokud se některá okna překrývají, text se vzájemně přepisuje. Obyčejně se pozice a rozměry oken volí tak, aby se nepřekrývaly.

Všechny výše popsané funkce (řízení kurzoru, vsouvání, mazání apod.) fungují v rámci aktuálně zvoleného okna - kurzor nemůže z okna "utéct". I obrazovkový editor (byl popsán dříve) dovoluje opravy pouze v rámci aktuálně zvoleného okna.

Řídicí kódy výstupu na obrazovku

Podprogram výstupu na obrazovku rozpoznává několik typů kódů:

- řídicí kódy (ØØ..1F)
- písmena velké a malé abecedy a další grafické znaky (kód ASCII - 2Ø..7F)
- znaky definovatelné uživatelem (8Ø..BF)
- česká a slovenská písmena s diakritickými znaménky v kódu KOI-8 čs. 2 (CØ..FF)

Kódy grafických znaků a písmen s diakritickým znaménkem odpovídají tabulce u klávesnice.

Řídicí kódy mají přiřazen význam podle následující tabulky.

Tabulka řídicích kódů výstupu na obrazovku

ØØ	NUL	nemá žádný význam
Ø1	ROW, n	nastaví kurzor na řádek číslo n, řádky se číslují od Ø a počítají se shora
Ø2	COLUMN, n	nastaví kurzor na sloupec číslo n, sloupce se číslují od Ø a počítají se zleva
Ø3	INVON	zapne inverzní zobrazení (černé na bílém)
Ø4	INVOFF	vrátí normální zobrazení (bílé na černém)
Ø5	PLEFT	nastaví kurzor na začátek logické řádky
Ø6	PRIGHT	nastaví kurzor na konec logické řádky
Ø7	BELL	způsobí pípnutí systémem definované výšky a délky
Ø8	BACKSPACE	vymaže znak vlevo od kurzoru

Ø9	TAB	vypíše mezery až do pozice dělitelné 8
ØA	LF	posune kurzor na další řádek (ve stejném sloupci)
ØB		nemá žádný význam
ØC		namá žádný význam
ØD	CRLF	nastaví kurzor na začátek následující řádky
ØE	CRSON	povolí blikání kurzoru
ØF	CRSOFF	zakáže blikání kurzoru
1Ø	SELLPR	nemá žádný význam
11	SELWINDOW, n	nastaví výstup do okna číslo n (Ø..)
12	SETWINDOW, XØ, YØ, RX, RY	nastaví parametry aktuálního okna XØ - počáteční sloupec (Ø..39) YØ - počáteční řádek (Ø..2Ø) RX - počet sloupců (1..4Ø-XØ) RY - počet řádek (1..21-YØ)
13	DELCHAR	vymaže znak pod kurzorem
14	INSCHAR	vsune mezeru na místo kurzoru
15	DELLINE	vymaže řádek, na kterém je kurzor (zbytek obrazovky se posune nahoru)
16	INSLINE	vsune prázdný řádek na místo řádky pod kurzorem (zbytek obrazovky se posune dolů)
17	UP	posune kurzor na předchozí řádek (ve stejném sloupci)
18	LEFT	posune kurzor doleva o jeden znak
19	RIGHT	posune kurzor doprava o jeden znak
1A	DOWN	totéž co LF (ØA), ale na posledním řádku neprovede nic
1B	ESCAPE, n	prefix speciálních funkcí
1C	HOME	nastaví kurzor na první znak první řádky
1D	CR	nastaví kurzor na první znak ve stejné řádce
1E	CLRLN	vymaže znaky od kurzoru do konce řádky
1F	CLRALL	vymaže znaky od kurzoru do konce okna

Speciální funkce za prefixem ESCAPE

Za prefixem ESCAPE (1B) mohou následovat tyto kódy:

'F'	FAST	nastaví režim FAST - vypne zobrazování VIDEO-RAM, program probíhá plnou rychlostí
'S'	SLOW	nastaví režim SLOW - zapne zobrazování VIDEO-RAM, běh programu je v tomto režimu asi 6x pomalejší
'I'	IOINIT	inicializuje výstupní proceduru - nastaví všechna okna přes celý displej, zruší případný inverzní mód, nastaví zobrazování 252 linek a provede HOME
'L', n	LINES	nastaví zobrazování n linek VIDEO-RAM.

Adresace displeje

Mikropočítač ONDRA má grafický displej. To znamená, že v jisté matici (320 x 255 bodů) lze každý bod buď rozsvítit (bílá) nebo zhasnout (černá). To je realizováno jako bitová mapa v paměti - každému bodu odpovídá jeden bit paměti. Tato bitová mapa je součástí hlavní paměti mikropočítače - je na konci paměti na adresách D800H .. FFFFH. Přiřazení adresy jednotlivým bodům matice je popsáno v následujícím textu.

Vždy 8 bodů vedle sebe představuje jeden bajt. Bod z této osmice nejvíce vlevo odpovídá bitu 7, bod nejvíce vpravo bitu 0. Bajt (osmice bodů) v levém horním rohu obrazovky má adresu FFFFH. Směrem doprava se adresa každého bajtu zmenšuje o 100H: FFFFH, FEFFH, FDFFH, FCFFH .. F8FFH. Souřadnice X ovlivňuje pouze vyšší část adresy.

Ve svislém směru je adresace poněkud složitější. V prvním přiblížení můžeme říci, že směrem shora dolů se adresa každého bajtu zmenšuje o 1, bajty v prvním sloupci mají adresy: FFFFH, FFFEH, FFFDH, FFFCH .. FF01H. Na obrazovce lze zobrazit 255 linek, adresa FF00H (a dále i FE00H, FD00H

.. D8ØØH) je nevyužita. Souřadnice Y ovlivňuje pouze nižší část adresy.

Takto jednoduché to ale bohužel není. Nižší část adresy je potřeba "pootočit" o jeden bit doprava: b7->b6, b6->b5, b5->b4 .. b1->bØ, bØ->b7.

Místo výše uvedených adres
FFFFH, FFFEH, FFFDH, FFFCH ..
je správně:
FFFFH, FF7FH, FFFEH, FF7EH ..

Pro výpočet adresy bodu na souřadnicích [X,Y] (bod [Ø,Ø] je vlevo nahoře) použijeme tyto vzorce:

```
AH = 255 - X/8
AL = RRA (255 -Y)
MASKA = 2 ** (7 - (X MOD 8))
```

AH je vyšší část adresy
AL je nižší část adresy
MASKA určuje bit va bajtu (8ØH, 4ØH, 2ØH .. 1)
RRA je výše popsaná operace pootočení doprava o bit
** je operace mocnění
MOD je operace, jejímž výsledkem je zbytek po dělení

Tabulka adres bajtů v displeji

```
FFFF FEFF FDFF ...    D8FF
FF7F FE7F FD7F ...    D87F
FFFE FEFE FDFE ...    D8FE
FF7E FE7E FD7E ...    D87E


FFØ2 FEØ2 FDØ2 ...    D8Ø2
FF81 FE81 FD81 ...    D881
FFØ1 FEØ1 FDØ1 ...    D8Ø1
FF8Ø FE8Ø FD8Ø ...    D88Ø
```

Po spuštění ONDRY je nastaveno zobrazení 252 linek. Nejvyšší tři linky se nezobrazují (maximálně lze zobrazit 255 linek). Počet zobrazovaných linek lze změnit řídícím kódem ve výstupní proceduře (ESCAPE "L" počet_linek). Nezobrazují se horní linky. První zobrazená linka je vždy na stejném místě obrazovky - při zmenšování počtu zobrazených linek se obraz posouvá nahoru.

Příklady adresování displeje

1) Podprogram pro adresaci bodu na obrazovce

Vstup: HL = souřadnice Y (Ø..254)
DE = souřadnice X (Ø..319)
Výstup: HL = adresa bajtu
A = maska (adresovaný bit je nastaven do "1")

```
ADDR:   PUSH    DE              ;uschovat obsah DE
        RRC     L               ;Y - pootočit doprava
        PUSH    HL              ;uschovat transformované Y
        LD      A,E             ;souřadnice X
        AND     ØØØØØ111B       ;vymaskovat číslo bitu
        LD      L,A             ;číslo bitu do L
        XOR     E               ;X s maskovaným číslem bitu
        SRA     D               ;bØ=> Carry, Ø => D
        RRA                     ;X podělit 8
        RRCA                    ;/4
        RRCA                    ;/8
        CPL                     ;255-X/8
        LD      DE,TABBIT       ;masky bitů
        LD      H,Ø             ;nulovat vyšší bajt indexu
        ADD     HL,DE           ;indexovat do tabulky
        POP     DE              ;nižší část adresy je v E
        LD      D,A             ;vyšší část adresy do D
        LD      A,(HL)          ;vybrat masku bitu z tabulky
        EX      DE,HL           ;adresa do HL
        POP     DE              ;vrátit původní obsah DE
        RET                     ;návrat z podprogramu
```

```
TABBIT: DB      80H,40H,20H,10H,8,4,2,1
```

2) Rozsvícení bodu v displeji

```
Vstup:  HL = souřadnice Y
        DE = souřadnice X
Výstup: rozsvícený bod na pozici [X,Y]

GSET:   CALL    ADDR            ;adresace bodu v displeji
        OR      (HL)            ;rozsvítit bod
        LD      (HL),A          ;zapsat nový stav
        RET                     ;návrat z podprogramu
```

3) Zhasnutí bodu v displeji

```
Vstup:  HL, DE = souřadnice bodu
Výstup: zhasnutý bod nma pozici [X,Y]

GRESET: CALL    ADDR            ;adresace bodu v displeji
        CPL                     ;negovat masku
        AND     (HL)            ;zhasnout příslušný bod
        LD      (HL),A          ;zapsat nový stav
        RET                     ;návrat z podprogramu
```

4) Adresace znaku v displeji (21 x 40 znaků, 12 x 8 bodů)

```
Vstup:  H = souřadnice Y (0..20)
        L = souřadnice X (0..39)
Výstup: HL = adresa spodní části znaku v displeji

SCRADR: LD      A,H             ;souřadnice Y
        ADD     A,A             ;vynásobit 6x
        LD      H,A             ;*2
        ADD     A,A             ;*4
        ADD     A,H             ;*6
        SUB     0F9H            ;speciální konstanta
        CPL                     ;negovat
        LD      H,L             ;souřadnici X do H
        LD      L,A             ;nižší část adresy je hotova
```

```
LD      A,H             ;souřadnice X
CPL                     ;negovat
LD      H,A             ;vyšší část adresy je hotova
RET                     ;návrat z podprogramu
```

Organizace paměti

Mikropočítač ONDRA má 64 KB pamět RAM a 4 KB paměti ROM. Paměť ROM obsahuje rezidentní programové vybavení - budeme ho honosně nazývat operační systém - OS. Část paměti RAM zabírá displej (1Ø KB), další část (2.75 KB) zabírá OS pro své potřeby. Zbytek (51 KB) je volný pro uživatelské programy.

Rozdělení paměti RAM

ØØØØ..ØØØ2	RST Ø	používá se pro body zastavení
ØØØ8..ØØØA	SYSTEM-CALL	vstupní vektor volání služeb operačního systému
ØØ2Ø..ØØ22	ROM-CALL	vstupní vektor volání podprogramů v paměti ROM
ØØ38..ØØ3A	INTERRUPT	vektor časového přerušení
ØØ66..ØØ68	NMI	vektor nemaskovatelného přerušení - tlačítko RESET
ØØ69..(HIMEM)	user area	volná paměť pro uživatelské programy
(HIMEM)		poslední bajt volné paměti
(SCRBUF)	ASCII-FILE	místo pro znakovou reprezentaci obrazovky
(TAPEBUF)	TAPE-BUFFER	místo pro jeden záznam souboru
(SPLOW)..(SPBAZE)		místo pro zásobník
	JP-FILE	RAM-vektory (volané z ROM)
	RAM-PROCEDURES	podprogramy ležící v RAM
D5AØ..D5FF	RET-FILE	RAM-vektory (volané z ROM)

D6ØØ..D6FF	SYS-FILE	systémové proměnné
D7ØØ..D7FF	ZERO-PAGE	stránka paměti povinně vyplněná nulami
D8ØØ..FFFF	SCREEN	zobrazovaná část paměti

ASCII-FILE je tabulka znakové reprezentace obrazovky. Každé znakové pozici na obrazovce odpovídá jeden bajt v tabulce, který obsahuje kód znaku vypsaného na dané pozici (KOI-8 čs 2 - bylo popsáno dříve). Bajty jsou ukládány po řádcích - první bajt v tabulce přísluší prvnímu znaku na první řádce, další bajt druhému znaku atd. Na pozicích, které byly smazány řídícím kódem CLRLN nebo CLRALL ve výstupní proceduře, je kód Ø. Tato tabulka je využívána podprogramy pro výstup na obrazovku a obrazovkovým editorem. Rozsah tabulky je inicializován na 21x4Ø bajtů.

TAPE-BUFFER je tabulka používaná při práci se soubory na magnetofonu. Při zápisu do souboru se do tabulky ukládají data předávaná hlavním programem. Při zaplnění tabulky se její obsah zapíše jako jeden blok záznamu na magnetofon. Při čtení ze souboru se do tabulky načte jeden blok záznamu a data jsou postupně předávána programu, který ze souboru čte. Velikost tabulky je 1 KB. Tabulka je také využívána hlavním programem operačního systému pro uložení řádky připravené obrazovkovým editorem.

(SPLOW)..(SPBAZE) je oblast vyhrazená pro 128 položek zásobníku. Tento rozsah postačuje pro většinu programů. Velikost této oblasti je možné změnit. Při volbě rozsahu místa pro zásobník nezapomeňte na rezervaci 8 položek pro obsluhu přerušení a na to, že zásobník musí být umístěn nad adresou 4ØØØH ! Jinak dojde při obsluze přerušení ke zhroucení programu.

JP-FILE je tabulka vektorů (instrukcí JP adresa) volaných z ROM a mířících opět do ROM. Jsou to volání podprogramů, které je možné nahradit jinými - pro rozšíření funkcí operačního systému.

RAM-PROCEDURES je oblast, ve které jsou umístěny podprogramy, které musí ležet v paměti RAM (SYSTEM-CALL, ROM-CALL, mapování paměti apod.).

RET-FILE je tabulka vektorů (instrukcí RET) volaných z ROM. Jsou na ně přivedeny významné body OS. Význam je stejný jako JP-FILE.

SYSTEM-FILE je oblast, ve které jsou umístěny systémové proměnné - čítače, ukazatele, přepínače...

ZERO-PAGE je stránka paměti vyplněná nulami. Je to dáno konstrukcí mikropočítače - nejvyšší dva bity (b7 a b6) bajtů této stránky se ještě zobrazují.

Oblasti JP-FILE, RET-FILE a SYSTEM-FILE budou podrobně popsány v samostatných kapitolách.

Mapování paměti

Mikroprocesor Z-80 použitý v ONDROVI umí adresovat 64 KB paměti. Tento rozsah má paměť RAM - adresní rozsah je tedy zcela vyčerpán. V ONDROVI je ale ještě paměť ROM s rozsahem 4 KB. Jak jí adresovat ?

Tento problém je řešen tím, že část paměti RAM lze odpojit a na její místo se připne paměť ROM. Tomuto konstrukčnímu řešení se říká mapování paměti.

V mikropočítači ONDRA se přepíná prvních 16 KB paměti.

S mapováním jsou spojeny určité programátorské problémy - mapování nelze provádět kdykoli. Nastávají tyto problémy:

- program ležící v prvních 16 KB paměti nesmí provádět mapování. Po přepnutí "zmizí" program v RAM a objeví se ROM s jiným programem, což samozřejmě vede ke zhroucení programu. Lze to přirovnat k odříznutí větve, na které sami sedíme
- podprogramy v ROM nelze zavolat pouhou instrukcí CALL. Je potřeba napřed provést přemapování na ROM a po ukončení podprogramu opět přemapovat na RAM a pokračovat v programu

Jaké je řešení těchto problémů ?

Nejjednodušší řešení je namapovat trvale ROM. Problémy s mapováním sice odpadají ale zbytečně se připravíme o 16 KB paměti RAM!

Jiným řešením je zkopírování obsahu ROM do RAM, ktará je pak trvale namapována. To nás připraví o 4 KB paměti RAM nebo při použití větších pamětí ROM (8 KB) o 16 KB, takže jsme, kde jsme byli.

Obě řešení jsou tedy polovičatá a použije je pouze nezkušený programátor.

Jaké je tedy správné řešení ?

Je třeba využít celou kapacitu pamětí RAM i ROM - musí se tedy průběžně mapovat podle potřeby.

Problémy s mapováním jsme vyřešili takto:

- mapování provádí pouze operační systém
- volání ROM probíhá jako volání služeb přes SYSTEM-CALL
- libovolný podprogram v ROM lze volat přes ROM-CALL

Jak probíhá volání podprogramu přes ROM-CALL ?

- v uživatelském programu je instrukce RST 2ØH za kterou následují dva bajty adresy podprogramu v ROM
- po provedení instrukce RST 2ØH se přes vektor na adrese 2ØH skočí do podprogramu v oblasti RAM-PROCEDURES (ve výpisu ROM se nazývá RROMCALL)
- vybere se adresa podprogramu za instrukcí RST 2ØH
- na zásobník se jako návratová adresa uloží adresa MAPRAM
- provede se namapování ROM (podprogram MAPROM)
- zavolá se podprogram v ROM
- po návratu na MAPRAM se přemapuje zpět na RAM
- provede se návrat do uživatelského programu za volání

I když tato služba umožňuje zavolat libovolný podprogram z ROM, je vhodné tuto službu použít pouze ve speciálním případu, kdy nevyhovuje žádná ze služeb SYSTEM-CALL. V naprosté většině případů služba SYSTEM-CALL plně vyhovuje.

Volání SYSTEM-CALL probíhá obdobně. Je použita instrukce RST 8 za kterou následuje číslo volané služby (jeden bajt). Popis služeb je v samostatné kapitole.

Toto volání má i další výhody:

- zavolání služby zabere pouze dva bajty
- volání služby číslem je nezávislé na verzi programu v ROM, je možné program v ROM předělat bez potřeby změnit i uživatelské programy

SYSTEM-CALL - volání služeb OS

Operační systém v ROM poskytuje určité funkce - služby. Tyto služby se volají pomocí instrukce RST 8 (kód ØCFH) za kterou následuje číslo služby (jeden bajt). Uživatel nepotřebuje znát adresy příslušných podprogramů v ROM ale pouze čísla služeb.

Tabulka služeb OS

číslo	název	funkce
Ø	PRINT	výstup znaku na obrazovku
1	LPRINT	výstup na tiskárnu
2	INKEY	čtení znaku z klávesnice
3	WAIT_INKEY	čtení znaku z klávesnice (s čekáním)
4	GET_SCAN_CODE	čtení kódu tlačítek klávesnice
5	EDIT_LINE	editace v jedné řádce
6	EDIT_SCREEN	editace po celé obrazovce
7	BEEP	pípnutí
8	OPEN_FILE	otevření souboru
9	WRITE_BYTE	výstup bajt do souboru
1Ø	READ_BYTE	čtení bajtu ze souboru

11	CLOSE_FILE	zavření souboru
12	TAPE_ON	zapnutí magnetofonu
13	TAPE_OFF	vypnutí magnetofonu
14	READ_ROM	načtení bajtu z ROM
15	IO_INIT	iniciace výstupní procedury
16	EXIT	ukončení programu
17	READ_ASCII	načtení znaku z displeje

V následujícím popisu služeb jsou vždy uvedeny tyto informace:

- číslo a název služby
- vstupní parametry (za slovem CALL) - ve kterých registrech se předávají při volání služby a jaký mají význam
- výstupní parametry (za slovem RET) - ve kterých registrech se vrací a jaký mají význam
- které registry služba změní (za slovem "ruší") - ve všech ostatních registrech je po návratu stejná hodnota jako při volání.
- popis funkce služby
- příklad použití

Znaky "xxx" znamenají, že se žádný vstupní (v odstavci CALL:) nebo výstupní (v odstavci RET:) parametry nepředávají nebo že se nemění obsah žádných registrů (v odstavci "ruší:").

Volání služeb pomocí instrukce RST 8 a DB čísla služby je jednoduché, úsporné a jasné, má však z hlediska ladění programů podstatnou nevýhodu - toto volání nejde krokovat bez speciálního programového vybavení! Proto je vhodnější napsat samostatné podprogramy pro všechny použité služby a v programu volat pouze tyto podprogramy instrukcí CALL, nikoliv přímo OS. Jedná se o nadefinování MACRO-instrukce SYS a doplnění potřebných podprogramů. Realizaci demonstruje následující příklad:

```
; Echo znaků napsaných na klávesnici na výstup
```

```
SYSVEC  EQU     8               ;vektor SYSTEM_CALL

SYS     MACRO   SLUZBA          ;definice MACRI-instrukce
        CALL    SLUZBA          ;volání služby instrukcí CALL
        ENDM                    ;konec definice MACRO

PRINT:  RST     SYSVEC          ;volání SYSTEM_CALL
        DB      1               ;služba PRINT
        RET                     ;návrat z podprogramu

WAIT_INKEY:
        RST     SYSVEC          ;volání SYSTEM_CALL
        DB      3               ;služba WAIT_INKEY
        RET                     ;návrat z podprogramu

; Hlavní program

ECHO:   SYS     WAIT_INKEY      ;vstup z klávesnice
        SYS     PRINT           ;výstup na displej
        JR      ECHO            ;nekonečná smyčka
```

Popis jednotlivých služeb OS

Ø PRINT

CALL: A = znak nebo řídící kód
RET: xxx
ruší: xxx

Výstup znaku na displej. Kódy byly popsány v samostatné kapitole.

```
; Výpis mezery na displej

PRSP:   LD      A,' '           ;dej do A kód mezery
        SYS     PRINT           ;volání služby PRINT
```

1 LPRINT

CALL: A = znak nebo řídící kód
RET: xxx
ruší: xxx

Výstup znaku na tiskárnu. OS definuje pouze výstupní vektor, neobsahuje vlastní komunikační podprogram. Tento podprogram je uveden ve výpisu ROM.

```
; Výpis mezery na tiskárnu

LPRSP:  LD      A,' '           ;dej do A kód mezery
        SYS     LPRINT          ;volání služby LPRINT
```

2 INKEY

CALL: xxx
RET: A = kód tlačítka (KOI 8 čs 2), Ø když není stisknuto žádné tlačítko
Příznak ZERO je nastaven, když A=Ø.
ruší: AF

Vrací kód stisknuté klávesy. Nečeká na stisknutí, když není žádné tlačítko stisknuto, vrací kód Ø.

```
; Test stisknutí klávesy CTRL C (přerušení běhu programu)

TCTRLC: SYS     INKEY           ;volání služby INKEY
        CP      'C'-'@'         ;test na kód klávesy CTRL C
        RET     NZ              ;návrat při jiné klávese
        JP      STOP            ;přerušení běhu programu
```

3 WAIT_INKEY

CALL: xxx
RET: A = kód tlačítka (KOI 8 čs 2)
ruší: AF

Čeká na stisknutí tlačítka. Při čekání zapíná zobrazování displeje (mód SLOW), po stisknutí tlačítka vrátí původní mód (SLOW nebo FAST).

```
; Čekání na stisknutí klávesy "nový_řádek"
```

```
GETCR:  SYS     WAIT_INKEY      ;volání služby WAIT_INKEY
        CP      NOVY_RADEK      ;test na kód klávesy
                                 "nový_řádek"
        JR      NZ,GETCR        ;když jiná klávesa - čekej
                                 dál
        RET                     ;návrat po stisknutí požado-
                                 vané klávesy
```

4 GET SCAN CODE

CALL: xxx

RET: A = kód stisknutého tlačítka (pořadové číslo)
A = Ø když není žádný kód připraven
Příznak ZERO je nastaven když A=Ø.

ruší: AF

Čte z tabulky kód stisknutého tlačítka - pořadové číslo v matici (1..5Ø). Když žádný kód v tabulce není, vrací Ø. Do tabulky se kódy zapisují při testu klávesnice při časovém přerušení (podprogram SCANKEY). Do tabulky se vejde 16 kódů. Kód se ukládá při stisknutí i při puštění tlačítka. Při puštění se k pořadovému číslu přičítá 128. U tlačítek použitých jako přeřaďovače se přičítá ještě 64.

Tabulka kódů tlačítek

	Ø	1	2	3	4
Ø	Ø1 R	Ø2 E	Ø3 W	Ø4 T	Ø5 Q
1	Ø6 F	Ø7 D	Ø8 S	Ø9 G	1Ø A
2	11 C	12 X	13 Z	14 V	15 ALT
3	16 SPACE				
4		22 Ø-9	23 ČS		25 SHIFT
5	26 J	27 K	28 L	29 B	3Ø NOVÝ_ŘÁDEK
6	31 U	32 I	33 O	34 Y	35 P
7	36 N	37 M	38 UP	39 B	4Ø CTRL
8		42 LEFT	43 DOWN		45 RIGHT
9	46 JV	47 JZ	48 JS	49 JJ	5Ø FIRE

Poslední řádek představuje křížový ovladač - směry JV = vprvo, JZ = vlevo, JS = nahoru, JJ = dolu a FIRE je tlačítko akce. Ze služby INKEY vrací stejné kódy jako šipky a mezerník.

Pomocí této služby lze sledovat aktuální stav tlačítek klávesnice - stisknuto/puštěno. Je potřeba si uvědomit, že konstrukce klávesnice umožňuje správné rozlišení maximálně dvou tlačítek držených najednou. Při stisknutí tří a více kláves najednou může být tento stav rozlišen správně, ale většinou budou vráceny kódy jiných tlačítek, než byly skutečně stisknuty! Toto je dáno konstrukcí a není možné programové ošetření této situace.

```
; Test stavu přeřaďovače CTRL

TCTRL:  SYS     GET_SCAN_CODE   ;volání služby GET_SCAN_CODE
        RET     Z               ;žádný kód není připraven
        LD      C,A             ;úschova kódu
        AND     ØØ111111B       ;vymaskovat pomocné bity
        CP      4Ø              ;test na kód klávesy CTRL
        RET     NZ              ;není to klávesa CTRL
        LD      A,C             ;načtený kód
        LD      (STAV_CTRL),A   ;úschova aktuálního stavu
                                 tlačítka CTRL
                                ;stisknuto při b7=1
        RET                     ;návrat z podprogramu
```

5 EDIT LINE

CALL: kurzor nastaven na editovaný řádek

RET: A = ukončovací kód

ruší: HL, DE, BC, AF

Editace jednoho řádku obrazovkovým editorem. Editace začne na pozici kurzoru. Návrat se provede po vypsání řídícího kódu, který podprogram editace řádky nezná. Tento kód se vrací v A. Podprogram

editace řádky rozpoznává tyto řídící kódy: vlevo, vpravo, na začátek/konec logické řádky, výmaz znaku, výmaz do konce řádky. Kódy nahoru a dolu ukončí editaci řádky pouze tehdy, když směřují pryč z editované řádky.

Vstupní řádek se nepřenáší do paměti, je třeba ho načíst z ASCII-FILE. Velikost editovaného řádku je omezena pouze velikostí aktuálního okna. Pro vstup řádky se obyčejně používá následující služba - EDIT_SCREEN

```
; Editace jedné řádky ukončená CR nebo ESCAPE

EDILIN: SYS     EDIT_LINE       ;volání služby EDIT_LINE
        CP      CR              ;test na klávesu CR
        RET     Z               ;ANO => návrat
        CP      ESCAPE          ;test na klávesu ESCAPE
        JR      NZ,EDILIN       ;NE => znovu editace
        SCF                     ;nastavit příznak ESCAPE
        RET     .               ;návrat z podprogramu
```

6 EDIT SCREEN

CALL: HL = adresa místa v paměti, kam se zapíše řádek po skončení editace

RET: HL = stejná hodnota jako při vstupu
Napsaný nebo opravený řádek je v paměti na zadané adrese.

ruší: DE, BC, AF

Editace v rámci celého okna. Funkce a možnosti obrazovkového editoru byly popsány dříve. Návrat se provede po stisku klávesy NOVÝ_ŘÁDEK. Jako výsledek se vrátí řádek, na kterém byl kurzor v okamžiku stisknutí klávesy NOVÝ ŘÁDEK. Text je ukončen kódem Ø.

Pokud chceme omezit pohyb kurzoru na určitou oblast na obrazovce, otevřeme okno na danou pozici. Obrazovkový editor umožní pohyb kurzoru pouze v tomto okně. Pokud chceme editovat text, vypíšeme ho

na displej, nastavíme na něj kurzor a zavoláme tuto službu. Kurzor je možné umístit i na pozici uprostřed řádky, nejenom na začátek - například na místo chyby ve špatně zadaném příkazu.

```
; Vstup řádky s možností editace po celé obrazovce

EDISCR: LD      HL,BUFFER       ;připravit do HL adresu místa v paměti pro napsaný řádek
        PUSH    DE              ;uschovat obsah DE
        PUSH    BC              ;uschovat obsah BC
        SYS     EDIT_SCREEN     ;volání služby EDIT_SCREEN
        POP     BC              ;vrátit původní hodnotu BC
        POP     DE              ; a DE
        RET                     ;návrat z podprogramu
                                ;v HL je adresa BUFFERu
```

7 BEEP

CALL: B = délka tónu (po 20 ms)
C = výška tónu (0..7) * 32, tedy bity 5,6,7

RET: xxx

ruší: HL, B, AF

Zapne příslušný tón na požadovanou dobu. Číslo tónu je v horních třech bitech registru C, spodních pět bitů musí být vynulováno! Tón 0 je pauza.

```
; Hra všech 7 tónů po 1 sekundě

MUSIC:  LD      C,32            ;1.tón
MUS1:   LD      B,50            ;50x20ms je 1 sekunda
        SYS     BEEP            ;volání služby BEEP
        LD      A,C             ;zahraný tón do A
        OR      A               ;test na tón 0 (pauza)
        RET     Z               ;zahrána celá melodie => návrat
        ADD     A,32            ;další tón
        LD      C,A             ;číslo dalšího tónu do C
```

```
        JR      MUS1            ;zahrát další tón
```

8 OPEN FILE

CALL: HL = adresuje název souboru
DE = adresuje typ souboru
B = různé příznaky

RET: xxx

ruší: HL, DE, BC, AF

Otevření souboru. Název i typ souborů byly popsány dříve. Oba texty končí kódem menším než 32, typicky Ø.

Jednotlivé bity v registru B mají následující význam:

bit	"1"	"Ø"
Ø	MOTOR ON/OFF	MOTOR ON
1	dlouhý zavaděč	krátký zavaděč
2	SERIAL I/O	TAPE

Bit Ø určuje, zda se po načtení nebo zápisu každého bloku má zastavovat magnetofon. To je nutné, pokud se čtená data rovnou zpracovávají (např. překlad programu přímo z kazety) nebo když se data pro zápis do souboru generují pomalu (např. překlad programu přímo do souboru). Pro čtení programů ve strojovém kódu (typ .KÓD) není třeba zastavovat magnetofon.

Bit 1 určuje při zápisu do souboru, zda se má před každým blokem dělat delší mezera. To je nutné, pokud se soubor bude číst se zastavováním magnetofonu (Bit Ø =1). Delší mezery mezi bloky jsou nutné pro rozjezd magnetofonu. Před prvním blokem souboru se dělá delší mezera vždy.

Bit 2 je příznak vstupu nebo výstupu do seriové linky. Nastaví se automaticky, pokud název začíná znakem #.