INTERPRETADOR: SECCOMP 2017

Este documento descreve uma arquitetura fictícia. O objetivo é implementar um emulador/interpretador que siga a especificação proposta. A especificação é bastante simplificada a fim de facilitar a didática do minicurso.

I. MEMÓRIA

Endereçamento: Endereços de 2 bytes (16 bits), ou seja, a memória é endereçada de #0000 (0) a #FFFF (65535). Cada endereço (bloco) da memória armazena 1 byte.
Tamanho: 64kB
Divisão: 32kB inferiores para ROM, 32kB superiores para uso geral.

Mapa da Memória RAM:

```
+---------------+= 0xFFFF (65535) Fim da Memória RAM
|               |
|   Uso Geral   |
|---------------|= 0x7FFF (32767) Início da Memória de uso geral
|               |
|     ROM       |
+---------------+= 0x0000 (0) Início da Memória RAM (onde fica a ROM)
```

II. TELA/DISPLAY

O display é de 64x64 pixels colorido, cada pixel é de 2 bytes e segue a escala RGB (#-RGB), 4 bits mais a esquerda não são usados. O display tem uma memória separada da RAM, é a memória de vídeo. Podendo ser representado por uma matriz de pixels.

III. TECLAS

O *hardware* possui um segmento de memória separado dedicado ao teclado, sempre que uma tecla é pressionada, seu bit vai para um. No exemplo abaixo, as teclas C e Z estão pressionadas e as demais não. Teclas numéricas não são consideradas (isso pode ser adaptado na implementação, se desejado). OBS: 26 bits são usados, um para cada caracter.

```
  A B C .... X Y Z
| 0  0  1 ....  0  0 1 |
```

IV. REGISTRADORES

A arquitetura possui 16 registradores de propósito geral (R0,R1,R2,R3,…,RF), o PC (contador de programa) e o SP (ponteiro de pilha). Todos armazenam valores numéricos inteiros de 2 bytes (16 bits). Dessa forma, a escala de valores suportados é de -32768 a 32767 (usa-se complemento de dois).

OBS: Nos registradores PC e SP valores negativos não são considerados. O PC deve ser inicializado em 0 (início da ROM).

V. INSTRUÇÕES

Todas as instruções tem 4 bytes (32 bits), onde:
1º byte: Código de Operação (*opcode*);
2º byte: Especifica os Registradores;
3º e 4º byte: Especificam um imediato (valor) ou endereço de memória.

CÓDIGO				**INSTRUÇÃO**
b1	**b2**	**b3**	**b4**	
0-	--	--	--	NOP
10	x-	nn	nn	LD Rx,nnnn
11	xy	nn	nn	LD Rx,Ry
12	x-	nn	nn	LD (nnnn),Rx
13	x-	nn	nn	LD Rx,(nnnn)
14	x-	nn	nn	LD Rx,'KEY' *1
20	--	nn	nn	JP (nnnn)
21	--	nn	nn	CALL(nnnn)
22	--	--	--	RET
30	xy	nn	nn	BEQ Rx,Ry,(nnnn)
31	xy	nn	nn	BGT Rx,Ry,(nnnn)
32	xy	nn	nn	BLT Rx,Ry,(nnnn)
40	xy	--	--	ADD Rx,Ry
41	xy	--	--	SUB Rx,Ry
42	xy	--	--	MUL Rx,Ry
43	xy	--	--	DIV Rx,Ry
50	xy	--	--	AND Rx,Ry
51	xy	--	--	OR Rx,Ry
52	x-	--	--	NOT Rx
60	--	--	--	CLS
61	xy	wh	c-	DRW Rx,Ry,Rw,Rh,Rc *2
7-	--	--	--	HALT

(*nnnn*) = endereço
nnnn = imediato (valor)
Obs: Um valor *nnnn* tem sua parte mais significativa no byte 3 (padrão big-endian).

*1 O código ascii de 'KEY' é armazenado em hexadecimal nos bytes 3 e 4, *nnnn.*
*2 DRW é uma exceção ao padrão, pois aloca registradores nos bytes 3 e 4.