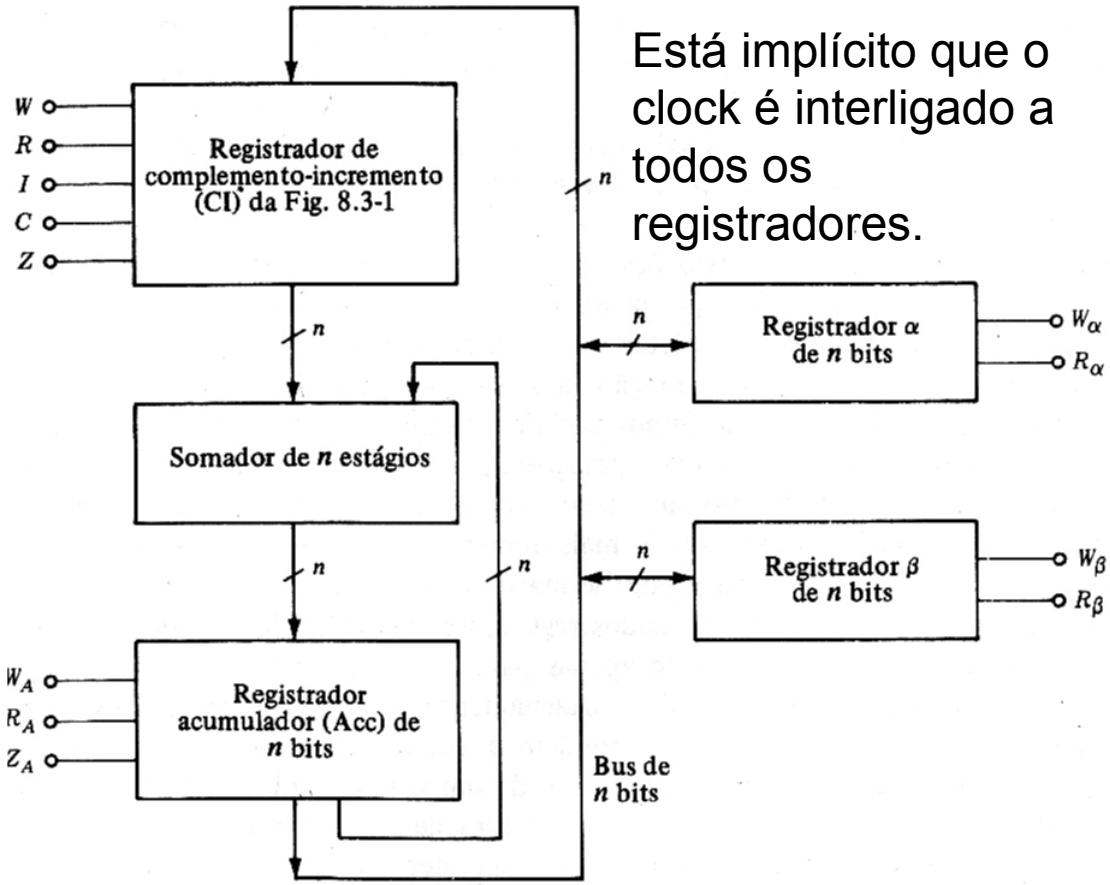
UNIVASF
Eletrônica Digital II

Computador – Simples e Aprimorado

Prof. Rodrigo Ramos
godoga@gmail.com

Registradores

- Exemplo de arquitetura simples:
  - Sistema para cálculo aritmético de conteúdo de dois registradores α e β



Está implícito que o clock é interligado a todos os registradores.

Sequência de comandos para calcular α + β

Ciclo	Terminal de controle ativado	Comentário
1	Z, Z_A	Limpa acumulador
2	R_α, W	Ler de α e escrever em CI
3	R, W_A	CI para Acc passando pelo somador
4	R_β, W	Ler de β e escrever em CI
5	R, W_A	β somado ao Acc
6	R_A, W_α	Acc para α

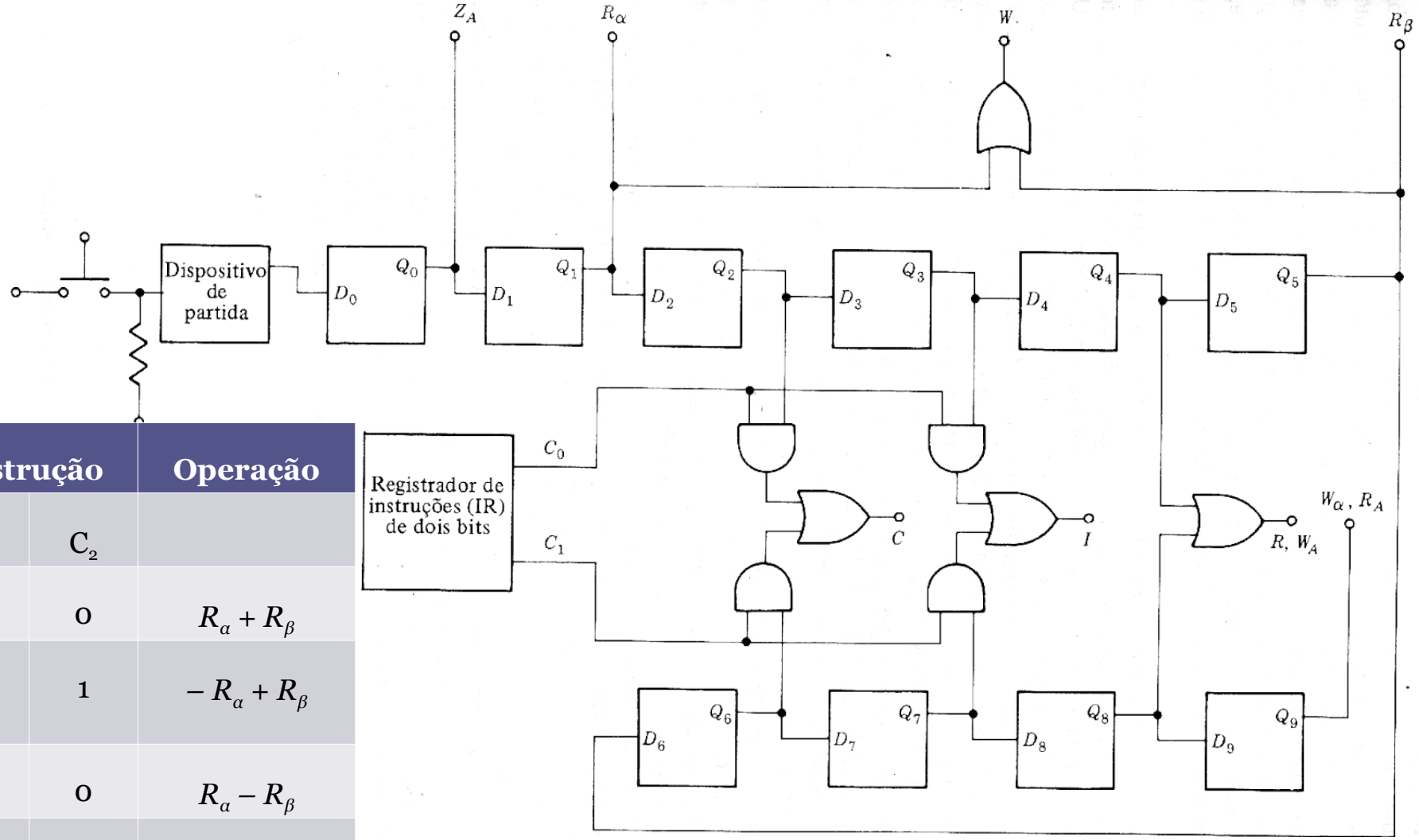
Controladores

- Sistema sequencial que fornece níveis lógicos temporizados para controlar operações lógicas simples que, juntas, executam operações mais complexas.

- Circuito fundamental para qualquer microprocessador.

Controladores

- Controlador para a arquitetura anterior.



Instrução		Operação
C_1	C_2	
0	0	$R_\alpha + R_\beta$
0	1	$-R_\alpha + R_\beta$
1	0	$R_\alpha - R_\beta$
1	1	$-R_\alpha - R_\beta$

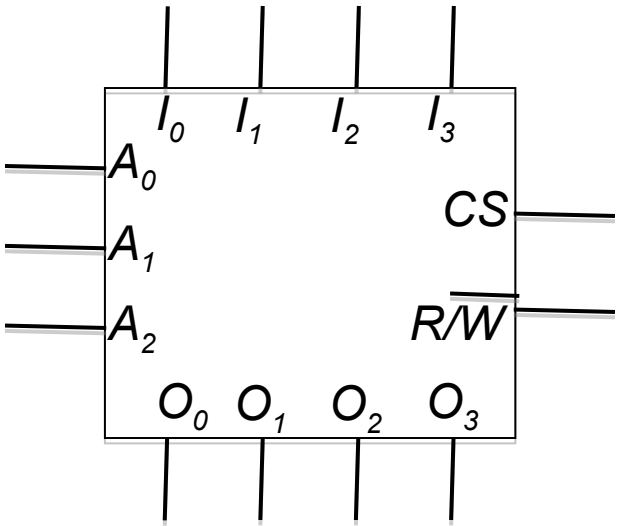
Computadores

- O sistema anterior é eficiente para operações com dois operandos (registradores α e β).

- Caso se necessite operar mais que dois números, deve-se ter uma arquitetura mais elaborada.
  - Substituição dos registradores por um banco de registradores (memória).
  - Operandos são submetidos às mesmas microoperações -> repetição da sequência para cada operando (a informação a respeito de como cada operando será calculado pode estar na memória)

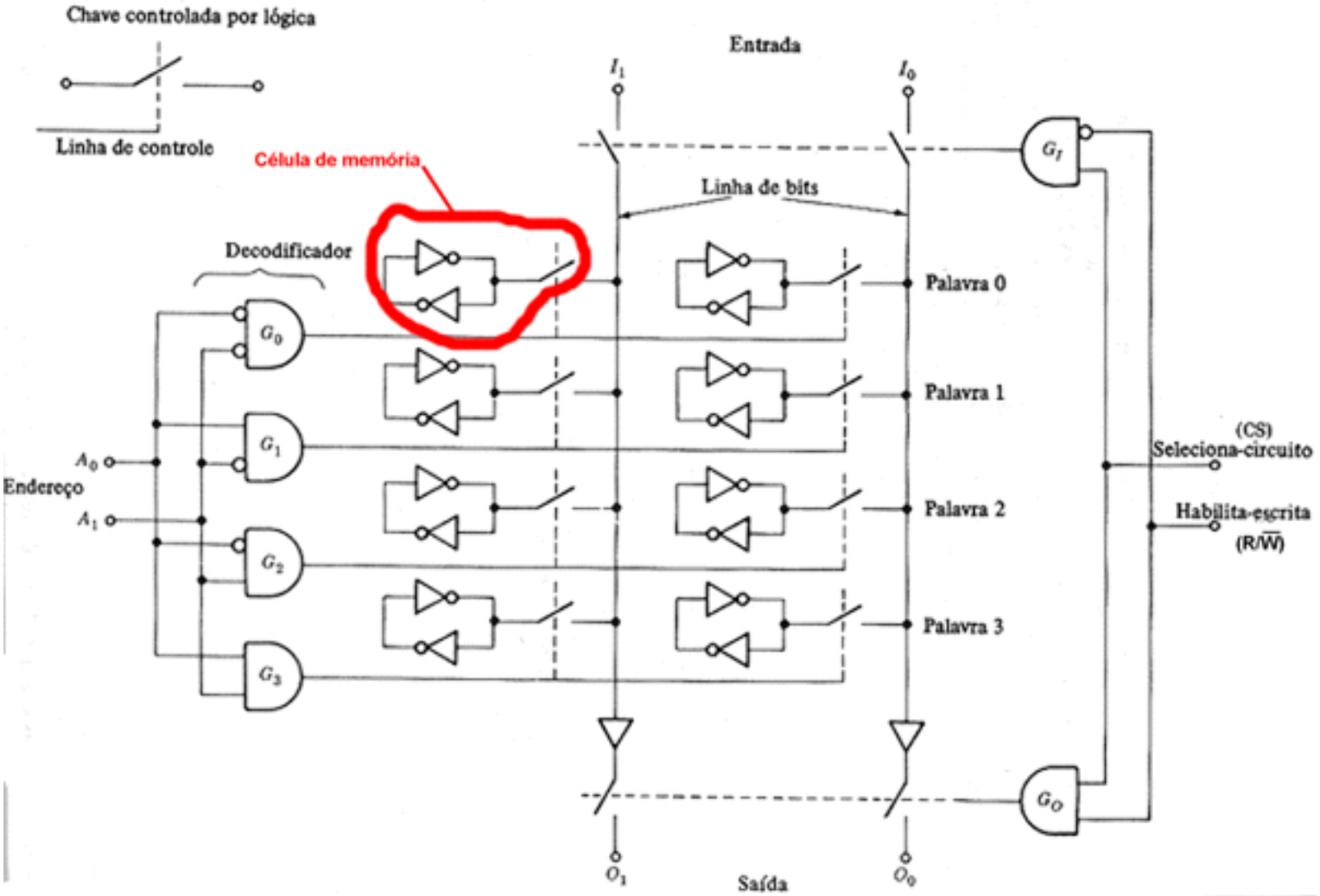
Memória RAM

- Memória de Acesso Aleatório
- Exemplo:
  - CI com oito palavras de 4 bits cada (oito registradores de 4 bits)
  - Registradores identificados pelo endereço $A_2 A_1 A_0$
  - Entrada de habilitação (*CS*)
  - Entrada de Leitura/Escrita ($R/\overline{W}$)



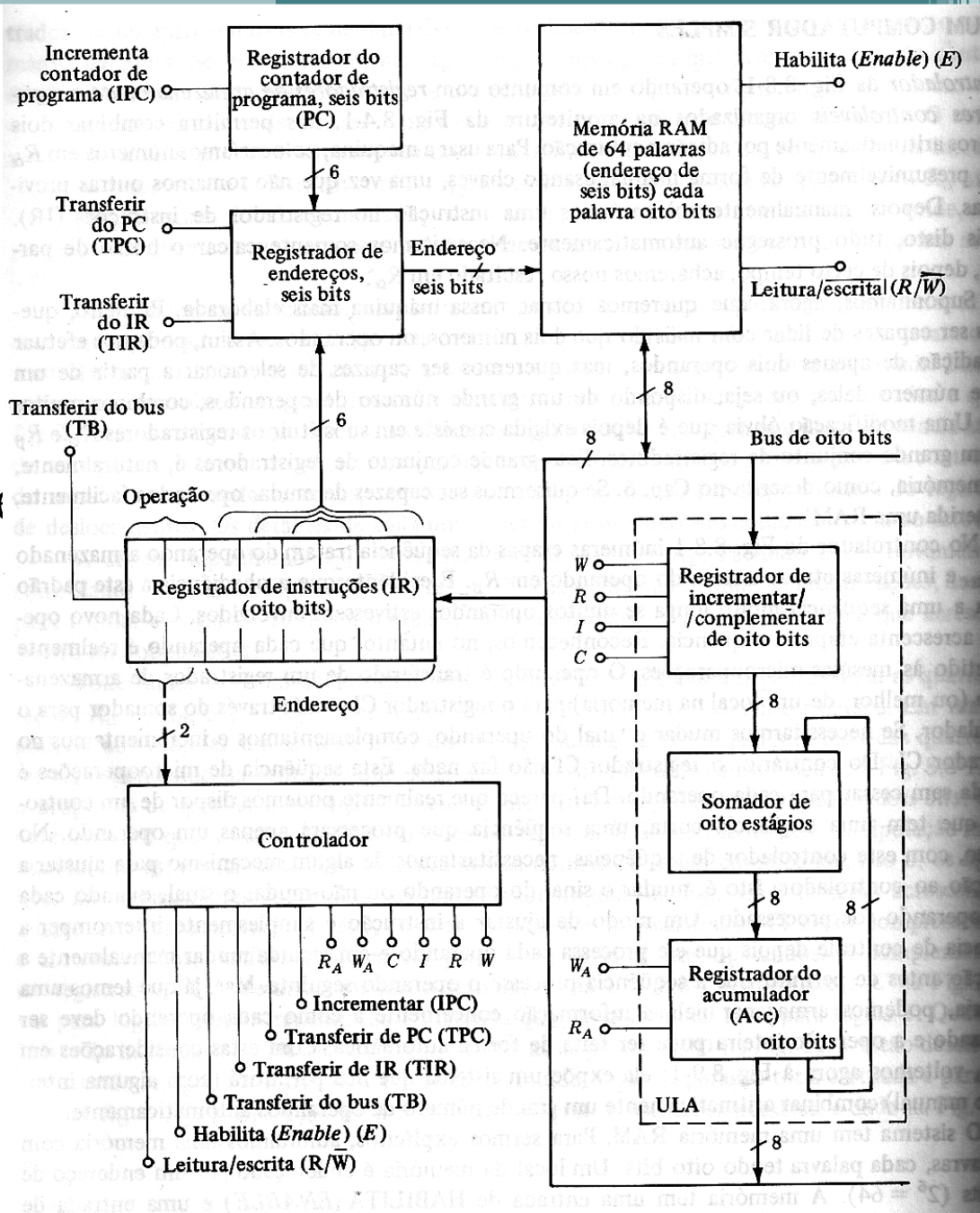
Memória RAM

- Estrutura interna



Computador Simples

- Considere o sistema que permite (com algumas intervenções manuais) combinar aritmeticamente um grande número de operandos.



Computador Simples

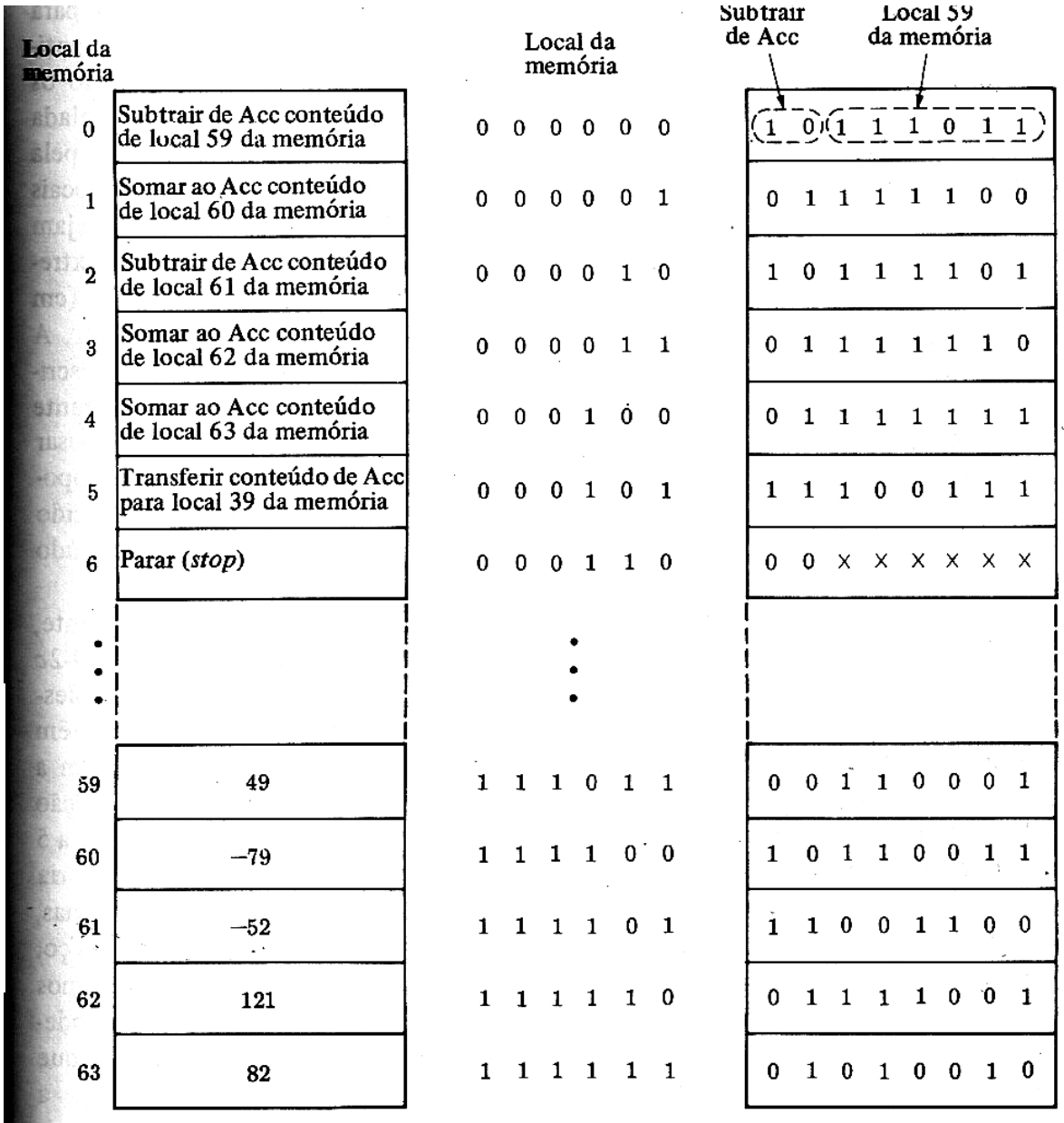
- Suponha o seguinte conteúdo da memória RAM e os códigos de instruções.
- Instrução: 2 bits + 6 bits

Observe que a memória armazena as instruções e os dados (operandos) em partes separadas.

Código	Instrução
00	Parar (*stop*)
01	Somar a Acc
10	Subtrair de Acc
11	Transferir conteúdo de Acc para

- Operação realizada:

(-49)+(-79)-(-52)+(121)+(82)



Computador Simples

- Operação
  - Suponha que o contador de programa (PC) e o acumulador estejam zerados

Ciclo de busca (*fetch cycle*): independe da instrução

Tabela 8.10-1 Ciclo de busca (fetch cycle)

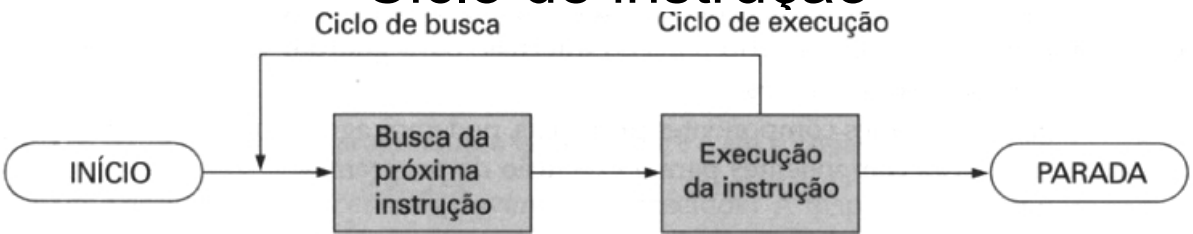
Ciclo de relógio	Descrição simbólica da operação	Linha de controle a ser habilitada
1. Transferir conteúdo do contador de programa para o registrador de endereços da memória	PC → MAR	TPC
2. Transferir instrução endereçada (no local 000000) para o registrador de instruções por (1) habilitação da memória para conectar a memória ao bus, (2) colocação de $R/\overline{W}$ em 1 para ler memória, e (3) transferência da palavra no bus para o registrador de instruções; incrementar contador de programa para preparar para fazer aparecer a próxima instrução quando a primeira instrução tiver sido concluída	M → IR ("M" representa palavra de memória endereçada) PC + 1 → PC	E, $R/\overline{W}$, TB IPC

Ciclo de execução

Tabela 8.10-2 Ciclo de executar

Ciclo de relógio	Descrição simbólica da operação	Linha de controle a ser habilitada
3. Transferir parte do endereço do registrador (seis bits à direita) para o registrador de endereços da memória (endereço é 59)	IR (ADD) → MAR	TIR
4. Transferir palavra endereçada da memória para o bus e do bus para o registrador CI	M → BUS BUS → CI	E, $R/\overline{W}$, W
5. Complementar CI	$\overline{CI}$ → CI	C
6. Incrementar CI	CI + 1 → CI	I
7. Registrar saída do somador no registrador do acumulador	Adder → Acc	W_A

Ciclo de instrução



Computador Simples

- Arquitetura e organização do computador
  - Quantidade e função dos registradores
  - Interligações entre registradores e memória
    - Frequentes referências a memória usando o MAR (acessado a partir do PC e do IR).
    - Computadores mais complexos fornecem acesso direto e indireto ao MAR).
  - Tamanho da memória
  - Tipos e operações da ULA
    - Operações de complemento, incremento e adição.
    - ULAs mais sofisticadas efetuam outras operações.

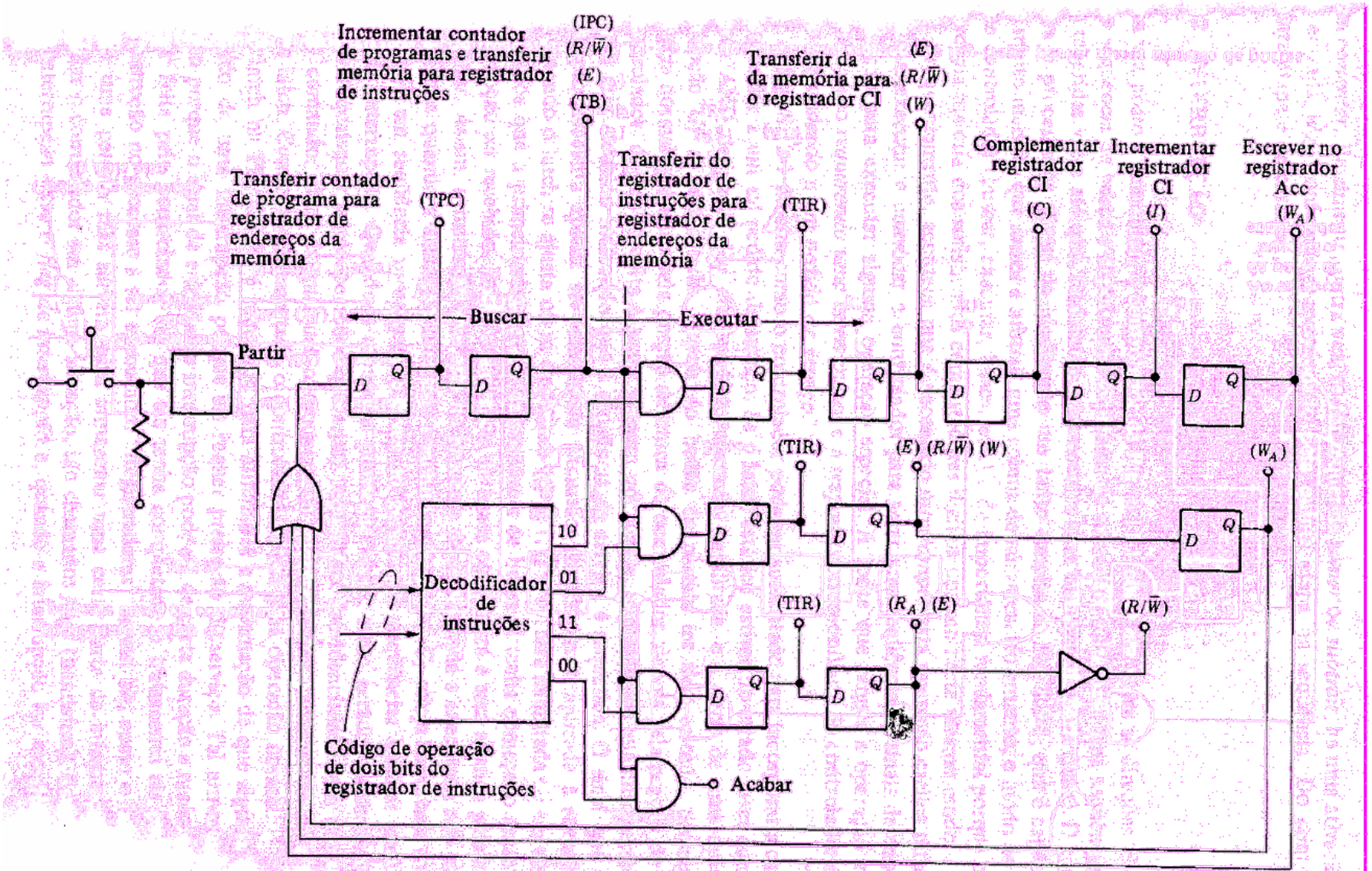
Computador Simples

- No projeto, foram implicitamente “entendidos” alguns aspectos
  - Na execução, fica “entendido” que a próxima instrução está no próximo local da memória.
  - Em uma operação, fica “entendido” que o outro operando encontra-se no acumulador.
  - Está “entendido” que o resultado de uma operação deve ficar armazenado no acumulador.

- Sem tais entendimentos, uma instrução teria que especificar:
  - Operação, fonte do primeiro operando, fonte do segundo operando, local de armazenamento e fonte da próxima instrução.
- Economia no comprimento das palavras e, consequentemente da memória e demais registradores.

Computador Simples

- Projeto do controlador



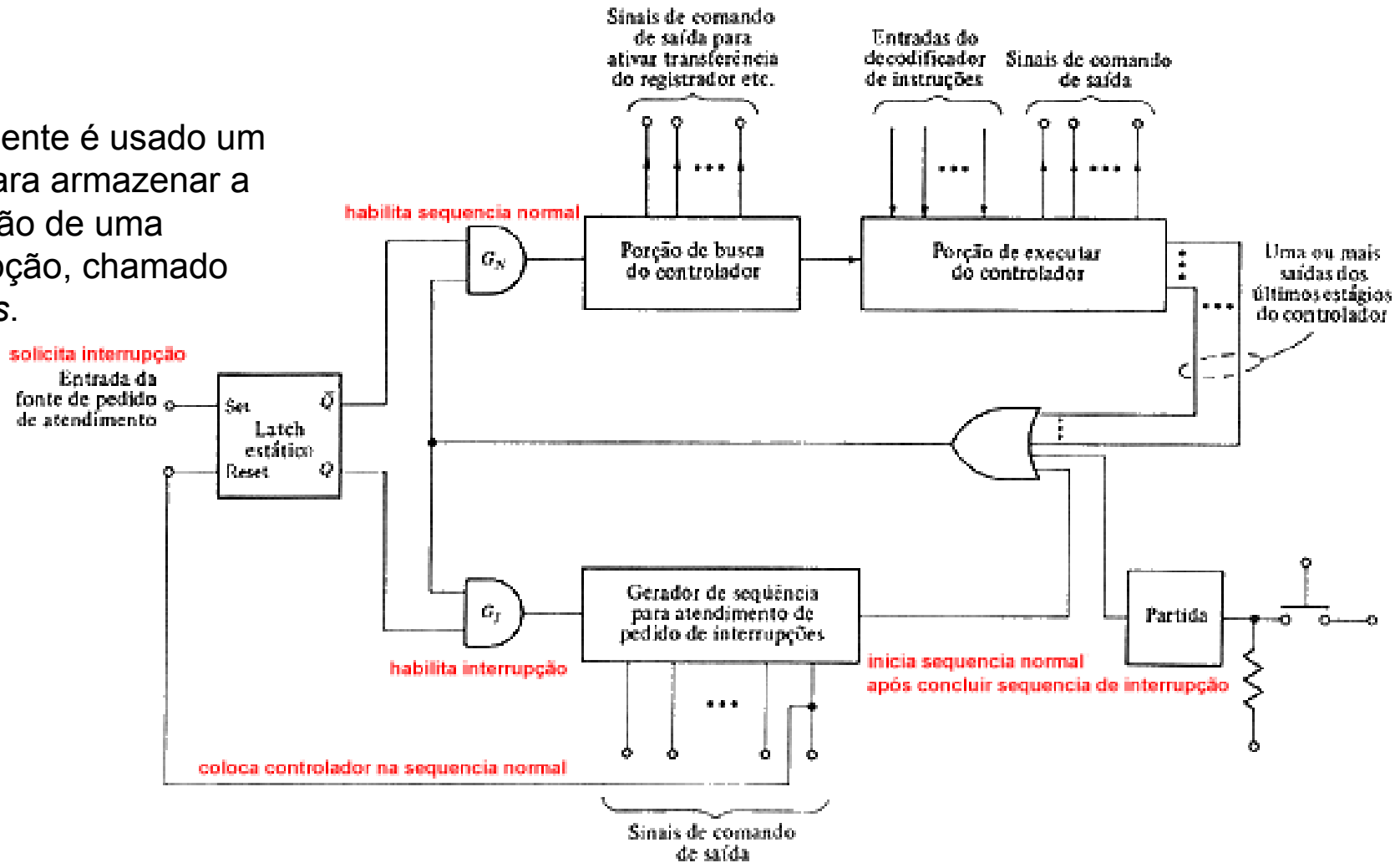
Computador Simples

- Interrupções
  - Frequentemente é necessário interromper o fluxo cíclico de operação de um computador para que o controlador execute outra sequência de comandos.
  - Esta parada é chamada de interrupção.
  - Classes de interrupção:
    - De software – gerada por resultado de instrução, ex. overflow em operação aritmética.
    - De relógio – gerada por relógio interno do processador.
    - De I/O – gerada por dispositivo de I/O para sinalizar conclusão de operação.
    - De falha de hardware – gerada na ocorrência de falha.

Computador Simples

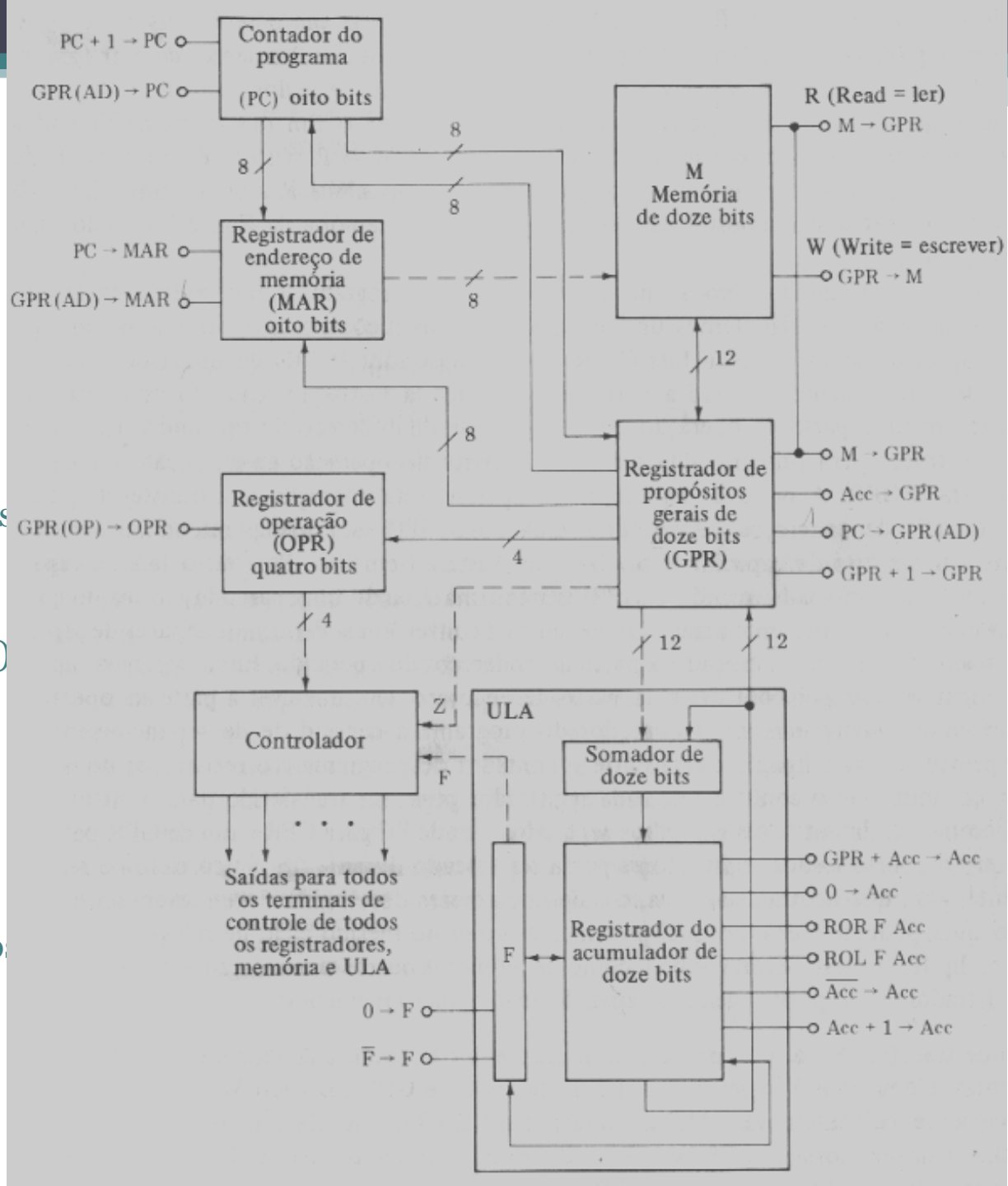
- Interrupções
  - Controlador modificado para receber interrupção

Geralmente é usado um lacth para armazenar a indicação de uma interrupção, chamado de *flags*.



Computador

- Arquitetura aprimorada
  - Memória de palavras de 12 bits
  - 16 instruções (4 bits de instrução)
  - Registrador de propósitos gerais (GPR)
  - Registrador de operações (OPR)
  - Conexão entre GPR e PC
  - ULA modificada
  - Linhas tracejadas indicam dados disponíveis independetes de ativação de controle



Computador

- Arquitetura aprimorada
- Componentes (exceto controlador) e operações de controle (18 microoperações)
- GPR(AD) refere-se à parte de endereço da instrução (8 bits) e GPR(OP) à parte do operando (4 bits)

Componente	Simbolismo de controle	Explicação
Memória	1. GPR → M	Escreve conteúdo de GPR na localização da memória endereçada
Contador do programa (PC)	2. PC + 1 → PC	Incrementa PC
	3. GPR(AD) → PC	Transfere bits de endereço do registrador de propósitos gerais para PC
Registrador de endereço da memória (MAR)	4. PC → MAR	Transfere do PC para MAR
	5. GPR(AD) → MAR	Transfere bits de endereço do registrador de propósitos gerais para MAR
Registrador de operação (OPR)	6. GPR(OP) → OPR	Transfere bits de operação do registrador de propósitos gerais para OPR
Registrador de propósitos gerais (GPR)	7. M → GPR	Transfere palavra endereçada para GPR
	8. Acc → GPR	Transfere conteudo do Acc para GPR
	9. PC → GPR(AD)	Transfere conteúdo do contador de programa para a parte de endereços do GPR
	10. GPR + 1 → GPR	Incrementa GPR
Unidade lógica e aritmética (ULA)	11. GPR + Acc → Acc	Adiciona número no GPR para número no Acc e deixa soma no Acc
	12. 0 → Acc	Limpa Acc
	13. ROR F, Acc	Gira Acc para a direita através de F
	14. ROL F, Acc	Gira Acc para a esquerda através de F
	15. 0 → F	Reset flip-flop F
	16. $\overline{F}$ → F	Complementa flip-flop F
	17. $\overline{Acc}$ → Acc	Complementa Acc
	18. Acc + 1 → Acc	Incrementa Acc

Computador

- Arquitetura aprimorada – Instruções de máquina
- É sem dúvida mais conveniente englobar microoperações de controle em instruções únicas, em que se usa mnemônicos.
- Veremos a seguir um conjunto de 16 instruções para o computador aprimorado.

Computador

- Arquitetura aprimorada - Instruções

- Ciclo de busca, assumindo que PC aponta para primeira instrução.

Ciclo de relógio	Microoperação	Explicação
1	PC → MAR	Transfere local da instrução do contador de programa para o registrador de endereço da memória
2	M → GPR PC + 1 → PC	Transfere palavra endereçada para o registrador de propósitos gerais; incrementa contador de programa
3	GPR (OP) → OPR	Transfere parte da operação de instrução para o registrador de operações

Computador

- Arquitetura aprimorada – Instruções
  - Instruções que ativam apenas um terminal de comando.

Instrução (microoperação)	Explicação	Mnemônico
$0 \rightarrow Acc$	Limpar acumulador	CRA
$\overline{Acc} \rightarrow Acc$	Complementar acumulador	CTA
$Acc + 1 \rightarrow Acc$	Incrementar acumulador	ITA
$0 \rightarrow F$	Limpar flip-flop F	CRF
$\overline{F} \rightarrow F$	Complementar flip-flop F	CTF
$PC + 1 \rightarrow PC$	Saltar (skip) para a instrução seguinte se F for zero	SFZ
Girar à direita	Girar à direita através de F e Acc	ROR
Girar à esquerda	Girar à esquerda através de F e Acc	ROL

Computador

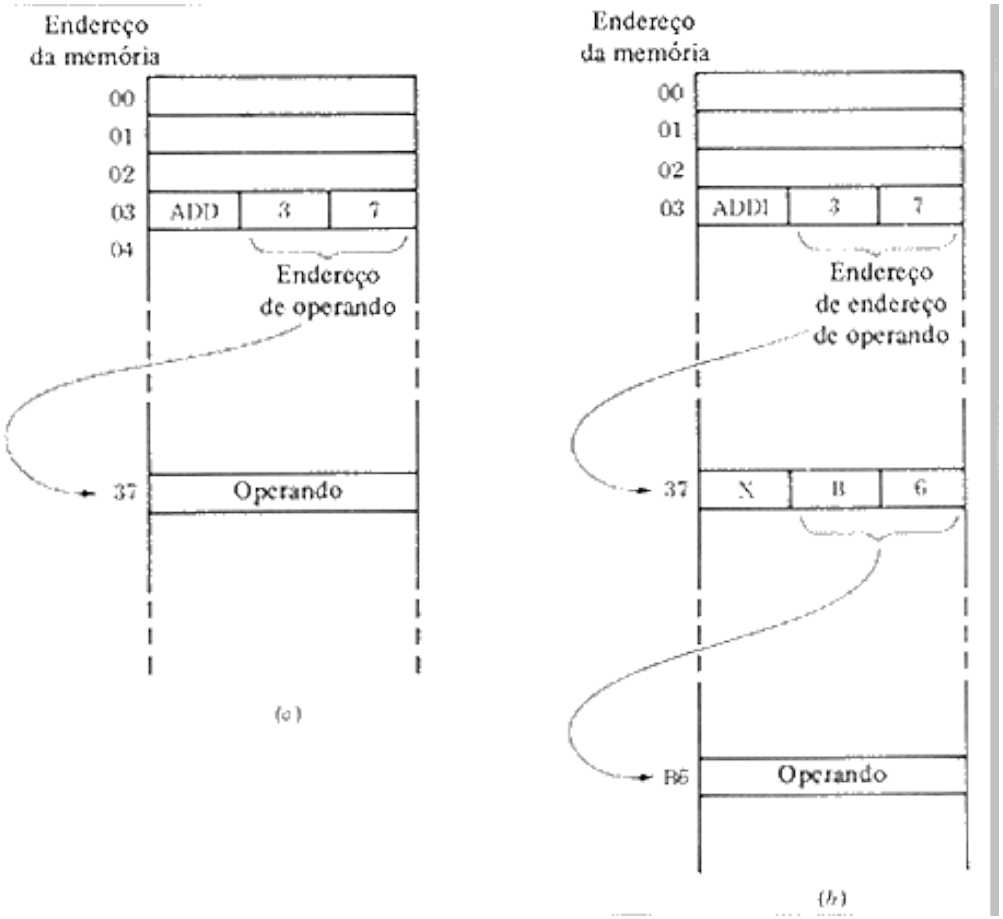
- Arquitetura aprimorada – Instruções
  - Instruções que requerem sequências de microoperações.

- ADD, end – adiciona acumulador operando localizado no endereço de memória end.

Ciclo de relógio	Microoperação	Explicação
1	GPR(AD) → MAR	Transfere endereço do operando de GPR (AD) para MAR
2	M → GPR	Lê da memória a palavra no local cujo endereço está em MAR
3	GPR + Acc → Acc	Adiciona conteúdo de GPR ao conteúdo de Acc, deixando a soma em Acc

Computador

- Arquitetura aprimorada – Instruções
  - Instruções que requerem sequências de microoperações.
- Endereçamento indireto – operando é um endereço que contém operando (usado em soma de vários operandos).



Computador

- Arquitetura aprimorada – Instruções
  - Instruções que requerem sequências de microoperações.

- ADDI, end – adiciona ao acumulador operando cujo endereço está localizado no endereço de memória end.

Ciclo de relógio	Microoperação	Explicação
1	GPR(AD) → MAR	Transfere endereço de GPR para MAR
2	M → GPR	Transfere conteúdo da memória no local endereçado para GPR (GPR terá então endereço do operando)
3	GPR(AD) → MAR	Transfere endereço do operando para MAR
4	M → GPR	Transfere operando endereçado para GPR
5	GPR + Acc → Acc	Adiciona conteúdo de GPR ao Acc

Computador

- Arquitetura aprimorada – Instruções
  - Instruções que requerem sequências de microoperações.

- STA, end – armazena conteúdo do acumulador no endereço de memória end.

Ciclo de relógio	Microoperação	Explicação
1	GPR(AD) → MAR	Transferir endereço de GPR para MAR
2	Acc → GPR	Transferir conteúdo de Acc para GPR
3	GPR → M	Escrever conteúdo de GPR na memória no endereço retido em MAR

Computador

- Arquitetura aprimorada – Instruções
  - Instruções que requerem sequências de microoperações.

- JMP, end – salta para instrução localizada em end.

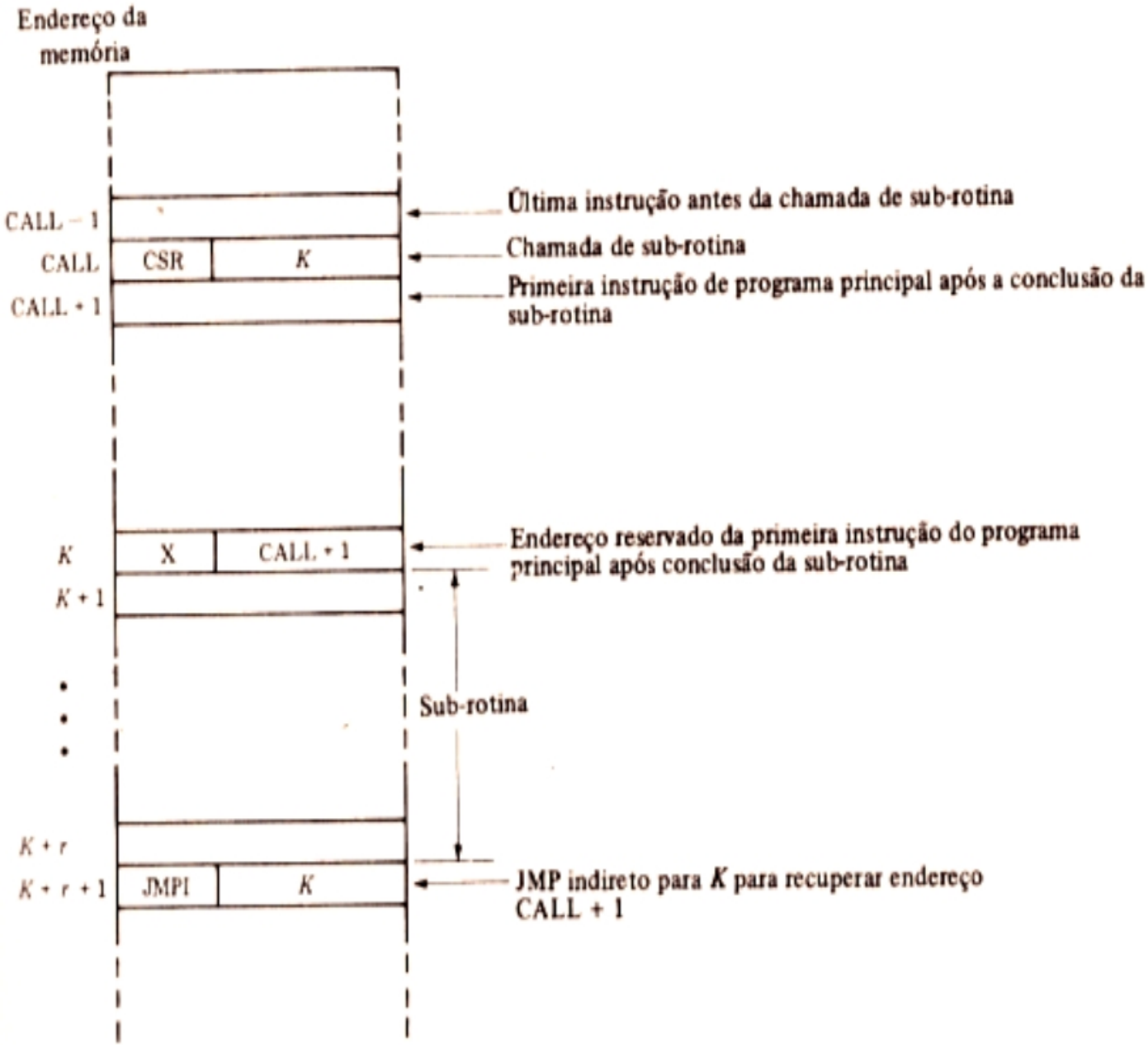
Ciclo de relógio	Microoperação	Explicação
1	GPR(AD) → PC	Transfere endereço da próxima instrução do GPR para PC

- JMPI, end – salta para instrução cujo endereço está localizada em end.

Ciclo de relógio	Microoperação	Explicação
1	GPR(AD) → MAR	Transfere endereço para MAR
2	M → GPR(AD)	Endereço lido da proxima instrução da memória
3	GPR(AD) → PC	Transfere endereço da próxima instrução do GPR para PC

Computador

- Arquitetura aprimorada – Instruções
  - Instruções que requerem sequências de microoperações.
- Chamadas de sub-rotina: trechos recorrentes de programas.



Computador

- Arquitetura aprimorada – Instruções
  - Instruções que requerem sequências de microoperações.
- Chamadas de sub-rotina: trechos recorrentes de programas.
- CSR, end – armazena endereço de retorno (CALL + 1) no local especificado por end (K) e busca próxima em K + 1.

Ciclo de relógio	Microoperação	Explicação
1	GPR(AD) → MAR	Transferir para MAR o endereço onde o endereço de retorno deve ser armazenado; este endereço de armazenamento é *K*
2	GPR(AD) → PC PC → GPR(AD)	Trocar os conteúdos de PC e GPR(AD) [após a troca, PC retém o endereço *K* e GPR(AD) retém CALL + 1, uma vez que PC foi incrementado de CALL para CALL + 1 durante o ciclo de busca]
3	GPR(AD) → M	Transferir GPR(AD) para a memória [o resultado é que endereço CALL + 1 será escrito no local *K* da memória]
4	PC + 1 → PC	Incrementar PC [PC guardará então endereço *K* + 1; próxima instrução será então tirada do local *K* + 1, que é a primeira instrução de sub-rotina

Computador

- Arquitetura aprimorada – Instruções
  - Instruções que requerem sequências de microoperações.

- ISZ, end – incrementar e saltar se zero. Ler o número em end, incrementá-lo e retorná-lo ao local original. Se zero, saltar próxima instrução.

Ciclo de relógio	Microoperações	Explicação
1	GPR (AD) → MAR	Transferir para MAR o local do número a ser incrementado
2	M → GPR	Ler número da memória
3	GPR + 1 → GPR	Incrementar número
4	GPR → M	Retornar número à memória
5	PC + 1 → PC (se GPR = 0)	Saltar próxima instrução se GPR = 0

Computador

- Arquitetura aprimorada – Instruções
- Conjunto de instruções, juntamente com arquitetura interna dos registradores, caracteriza um microprocessador.
- É importante salientar que a memória RAM não está presente nos microprocessadores.
- Um programa escrito na forma de mnemônicos é conhecido como Assembly. É necessário um montador (assembler) para conversão em código de máquina (bits).

Computador

- Arquitetura aprimorada – Instruções

1. CRA – limpar acumulador
2. CTA – complementar acumulador
3. ITA – incrementar acumulador
4. CRF – limpar F
5. CTF – complementar F
6. SFZ – saltar próxima instrução se F = 0
7. ROR – girar acumulador à direita com F
8. ROL – girar acumulador à esquerda com F
9. ADD end – adicionar ao acumulador conteúdo de end
10. ADDI end – adicionar ao acumulador conteúdo do conteúdo de end
11. STA end – armazenar em end conteúdo do acumulador
12. JMP end – saltar para end
13. JMPI end – saltar para posição cujo endereço está em end
14. CSR end – chamar sub-rotina em end
15. ISZ end – incrementar e saltar se Z = 0

- HLT - parar

Computador

- Arquitetura aprimorada
- Exemplo: Adição de três números.

Local da memória				Explicação
00	CRA	X	X	Limpar Acc
01	ADD	0	6	Adicionar conteúdo de 06 no Acc
02	ADD	0	7	Adicionar conteúdo de 07 no Acc
03	ADD	0	8	Adicionar conteúdo de 08 no Acc
04	STA	0	9	Armazenar conteúdo de Acc no local 09
05	HLT	X	X	Parar
06	0	1	7	Operando em 06
07	0	0	B	Operando em 07
08	0	1	C	Operando em 08
09				→ 03B será armazenado aqui no fim do programa

No caso geral, é conveniente especificar endereços em forma simbólica.

O montador se responsabiliza pela atribuição de endereço.

Computador

- Arquitetura aprimorada
- Exemplo: Adição de três números localizados nas posições W, X e Y da memória, e armazenar em Z.

Local simbólico	Conteúdo do local	Comentário
	CRA	Limpar Acc
	ADD W	Adicionar conteúdo de W a Acc
	ADD X	Adicionar conteúdo de X a Acc
	ADD Y	Adicionar conteúdo de Y a Acc
	STA Z	Armazenar conteúdo de Acc no local Z
	HLT	Parar
W	017	Operando em W
X	00B	Operando em X
Y	01C	Operando em Y
Z	XXX	Local de armazenamento do resultado

Computador

- Arquitetura aprimorada
- Exemplo: Programa para subtração.

Rótulo (Label)	Conteúdo	Comentário
	CRA	Limpar Acc
	ADD SUB	Adicionar subtraendo a Acc
	CTA	Complementar Acc
	ITA	Incrementar Acc
	ADD MIN	Adicionar minuendo a Acc
	STA DIF	Armazenar em local rotulado "DIF"
	HLT	Parar
SUB	09C	Subtraendo em local rotulado "SUB"
MIN	0B7	Minuendo em local rotulado "MIN"
DIF	XXX	Local onde a diferença será armazenada (a diferença será 0B7 – 09C = 01B)

Computador

- Arquitetura aprimorada
- Exemplo: Uso de JMP, ISZ e endereçamento indireto – Somar 100 parcelas localizadas sequencialmente na memória.

Rótulo (Label)	Conteúdo	Comentário
	CRA	Limpar Acc
LOOP	ADDI ANA	Adicionar ao Acc o número cujo endereço está no local "ANA" (endereço do próximo adendo)
	ISZ ANA	Incrementar endereço de adendo
	ISZ CTR	Incrementar número em local "CTR" e saltar próxima instrução se incremento faz o conteúdo de CTR igual a 0
	JMP LOOP	JMP de volta à instrução rotulada "LOOP"
	STA RES	Armazenar resultado em local "RES"
	HLT	Parar
RES	XXX	Resultado a ser armazenado aqui
ANA	FAD	Este local guarda endereço do próximo adendo
CTR	F9C	Este local guarda contagem de número de adições a serem feitas (representação de – 100 em complemento de dois)
FAD	ADD (1)	100 adendos a serem somados
	ADD (2)	
	.	
	.	
	.	
	ADD (100)	

Computador

- Arquitetura aprimorada
- Exemplo: Programa para multiplicação de dois números binários de 4 bits (resultado é um número de 8 bits).

Rótulo (*Label*)	Conteúdos	Comentário
	CRA STA SP	Limpar local onde a soma dos produtos SP, isto é, resultado da multiplicação, será armazenada
LOOP	ADD MR	Carregar o conteúdo do local MR em Acc
	ROR	Girar à direita para deslocar bit mais à direita do multiplicador dentro de F
	STA MR	Colocar multiplicador deslocado de volta ao local MR
	SFZ	Saltar (skip) próxima instrução se F = 0
	JMP 1	Saltar (jump) para instrução em local rotulado "1" (desde que F ≠ 0)
	JMP 0	Saltar (jump) para instrução em local rotulado "0" (desde que F = 0)
1	CRA ADD MD	Carregar conteúdo de local MD em Acc
	ADD SP	Adicionar conteúdo do local SP a Acc
	STA SP	Armazenar conteúdo de Acc no local SP
	CRF	Limpar flip-flop F
0	CRA ADD MD	Carregar conteúdo do local MD em Acc
	ROL	Girar à esquerda
	STA MD	Colocar multiplicando deslocado de volta ao local MD
	ISZ CTR	Incrementar contador
	JMP LOOP	Contador não zero, saltar (jump) de volta para instrução em "LOOP"
	HLT	Parar
CTR	FF4	Representação hexadecimal de −12
MD	Multiplicando	Local para guardar multiplicando
MR	Multiplicador	Local para reter multiplicador
SP	XXX	Local para reter resultado (limpo pelas duas primeiras instruções)

Computador

- Arquitetura aprimorada

- Exemplo: Programa para cálculo da operação $N_1N_2 + N_3N_4$. Como há multiplicações repetidas, pode-se usar sub-rotina.

Rótulo (*Label*)	Conteúdo	Comentário
A	N_1	Locais *A*, *B*, *C* e *D* da memória retêm números N_1, N_2, N_3 e N_4 a serem combinados em $N_1N_2 + N_3N_4$
B	N_2	
C	N_3	
D	N_4	
T	FF4	Representação hexadecimal de -12
	CRA ADD A STA MD	Coloca N_1 no local "MD", reservado ao multiplicando na sub-rotina da multiplicação
	CRA ADD B STA MR	Coloca N_2 no local "MR", reservado ao multiplicador na sub-rotina da multiplicação
	CSR MULT	Chama sub-rotina no endereço "MULT"
	CRA STA PR	Limpa local "PR" a ser usado para armazenar o resultado parcial N_1N_2
	ADD SP STA PR	Armazena resultado parcial N_1N_2 no local "PR"
	CRA ADD C STA MD CRA ADD D STA MR	Coloca N_3 e N_4 em "MD" e "MR", onde eles são acessíveis para sub-rotina da multiplicação
	CSR MULT	Chama novamente sub-rotina da multiplicação
	CRA ADD SP	Carrega Acc com N_3N_4 do local "SP"
	ADD PR	Adiciona N_1N_2 do local "PR" para formar $N_1N_2 + N_3N_4$
	HLT	Pára, deixando $N_1N_2 + N_3N_4$ em Acc

MULT		Retém endereço de retorno (mutável)
	CRA ADD T STA CTR	Coloca -12 no local da memória rotulado "CTR"
	CRA STA SP	Sub-rotina da multiplicação
LOOP	ADD MR ROR STA MR SFZ JMP 1 JMP 0	
1	CRA ADD MD ADD SP STA SP	
	Continuação da sub-rotina da multiplicação	
	CRF	
0	CRA ADD MD ROL STA MD ISZ CTR JMP LOOP	
	JMPI MULT	Retorna ao programa principal
CTR MD MR SP PR		Locais da memória rotulados reservados ao contador, multiplicando, multiplicador, soma de produtos parciais e aos resultados parciais