Microprocessadores e Microcontroladores (27146)

Microarquitetura

Prof. Ricardo Menotti (menotti@ufscar.br)

Atualizado em: 26 de abril de 2021

Departamento de Computação
Centro de Ciências Exatas e de Tecnologia
Universidade Federal de São Carlos

Conteúdo

Considerações iniciais

Codificação das instruções

Processador monociclo

Processador multiciclo

Processador pipeline

Referências

Considerações iniciais

Subconjunto das instruções implementado [Harris and Harris(2016)]

- Processamento de dados: `ADD`, `SUB`, `AND` e `ORR` (com operandos em registradores e imediatos, mas sem deslocamentos);
- Acesso à memória: `LDR` e `STR` (com `offset` imediato positivo).
- Salto: `B`.

Elementos de estado do processador ARM

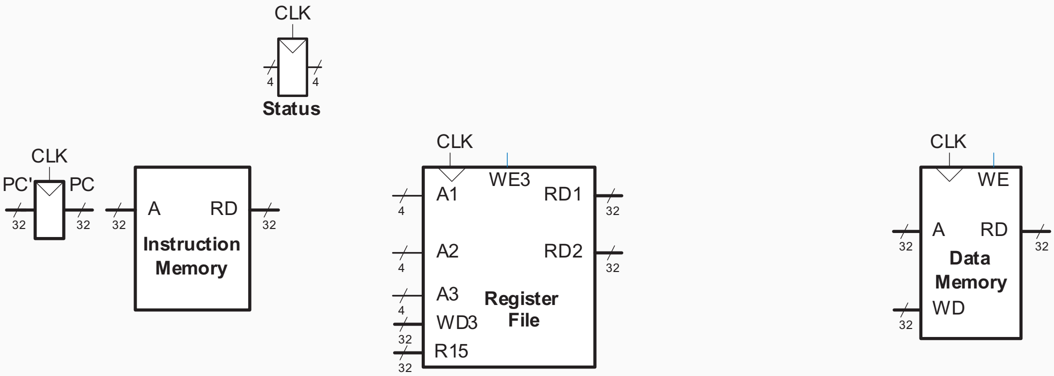


Figure 7.1 State elements of ARM processor

Tempo de execução de um programa

$$Execution\ Time = \left(\#instructions\right)\left(\frac{cycles}{instruction}\right)\left(\frac{seconds}{cycle}\right)$$

Codificação das instruções

Processamento de dados ($Op = 00$)

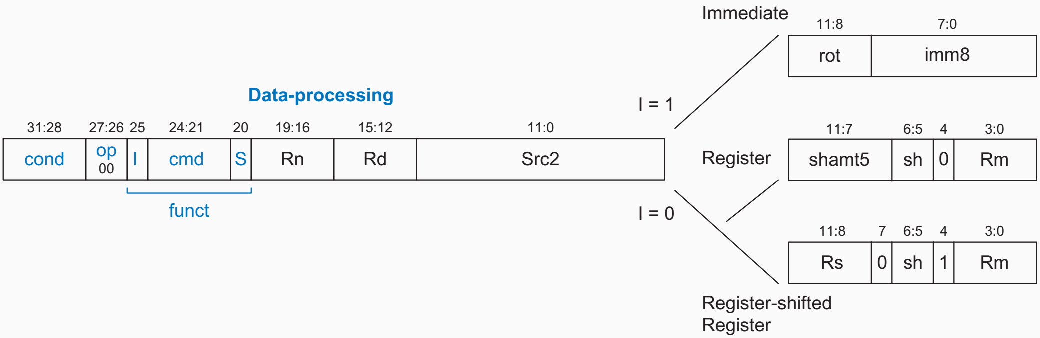


Figure 6.17 **Data-processing instruction format showing the *funct* field and *Src2* variations**

Acesso à memória ($Op = 01$)

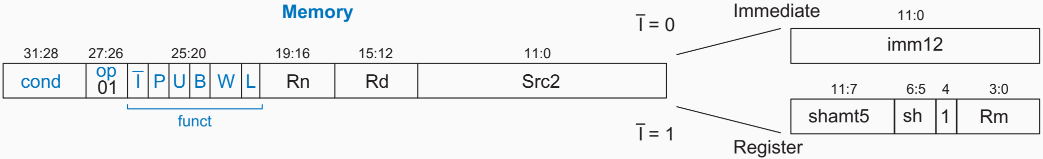
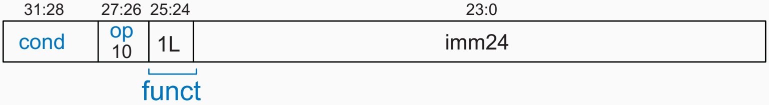


Figure 6.22 **Memory instruction format for** `LDR`**,** `STR`**,** `LDRB`**, and** `STRB`

Desvios ($Op = 10$)



31:28	27:26	25:24	23:0
cond	op 10	1L	imm24

Figure 6.24 **Branch instruction format**

Processador monociclo

Execução da instrução LDR

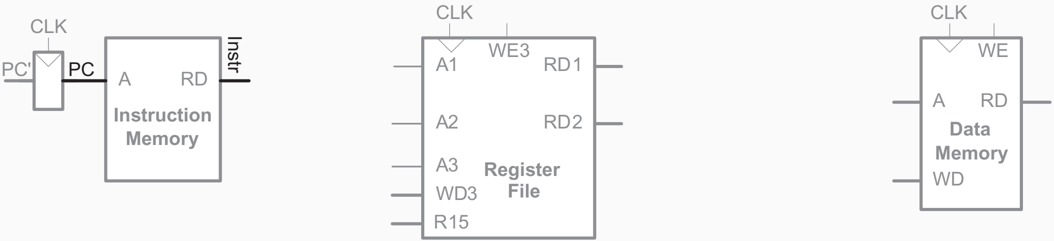


Figure 7.2 Fetch instruction from memory

Execução da instrução LDR

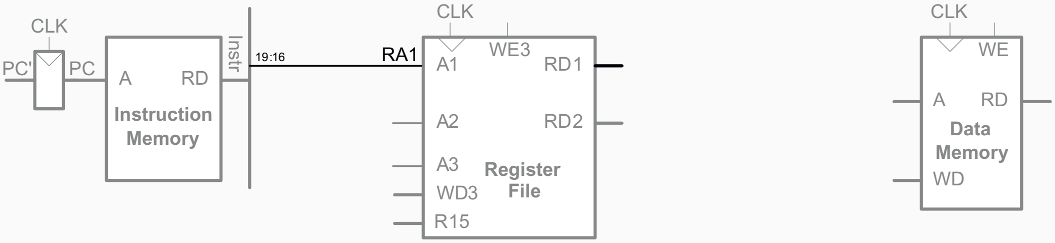


Figure 7.3 Read source operand from register file

Execução da instrução LDR

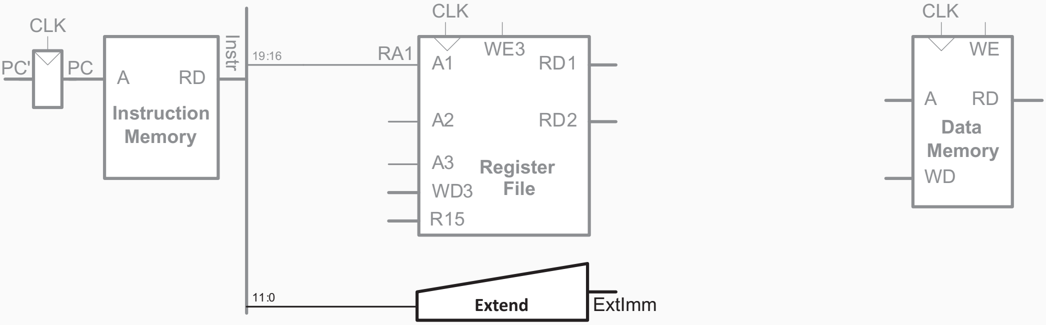


Figure 7.4 **Zero-extend the immediate**

Execução da instrução LDR

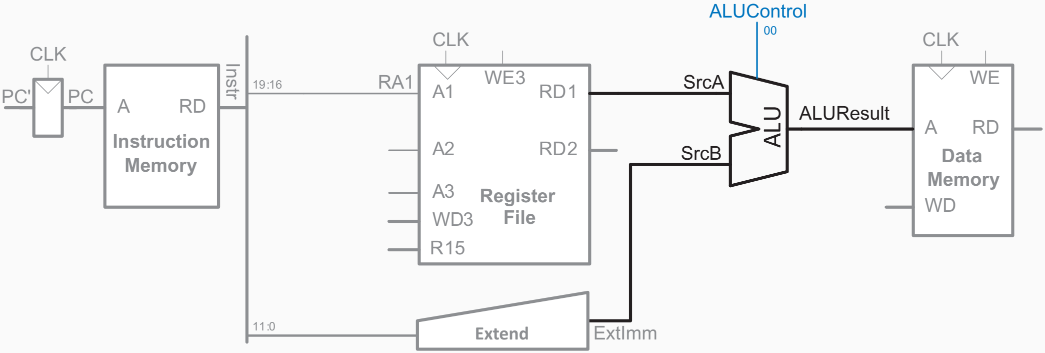


Figure 7.5 Compute memory address

Execução da instrução LDR

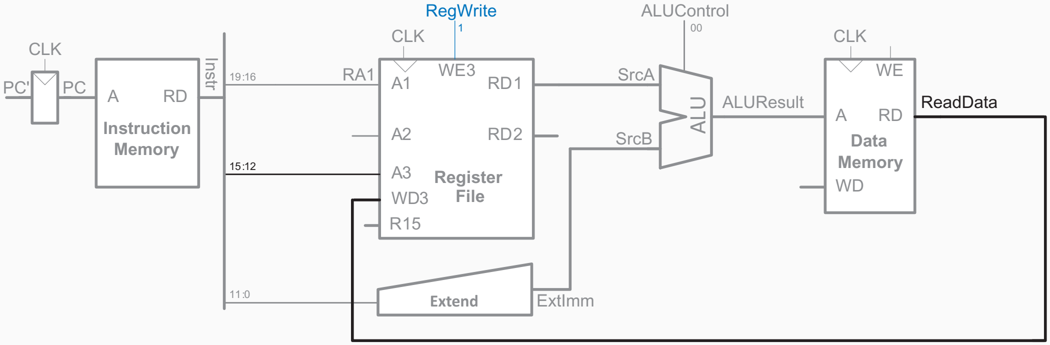


Figure 7.6 **Write data back to register file**

Execução da instrução LDR

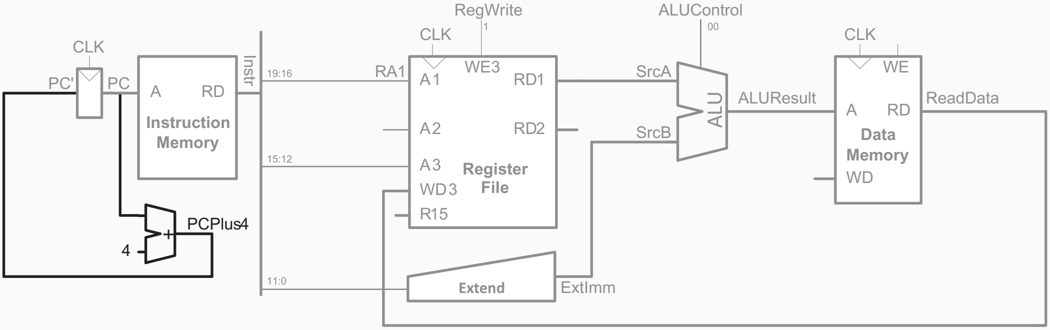


Figure 7.7 **Increment program counter**

Execução da instrução LDR

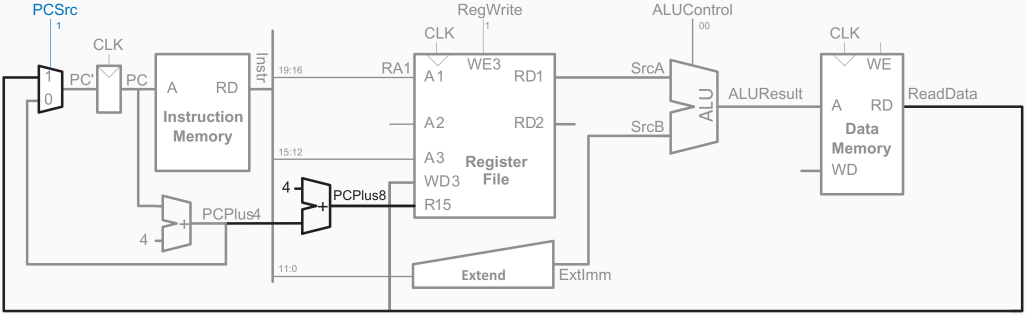


Figure 7.8 Read or write program counter as R15

Execução da instrução STR

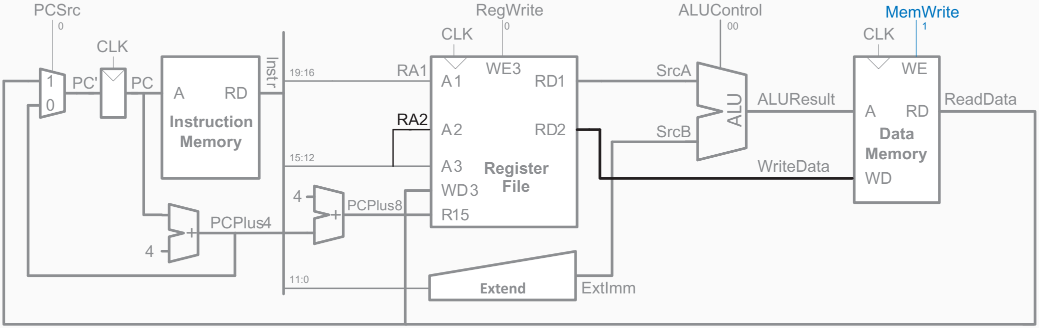


Figure 7.9 **Write data to memory for** STR **instruction**

Execução das instruções de processamento de dados com imediatos

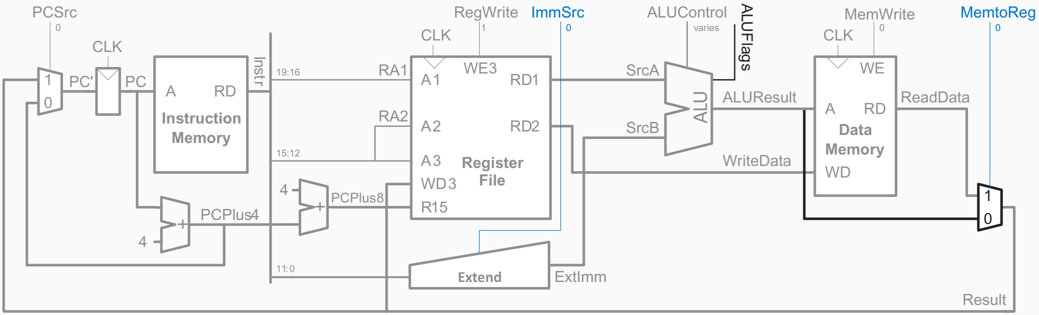


Figure 7.10 Datapath enhancements for data-processing instructions with immediate addressing

Execução das instruções de processamento de dados com registradores

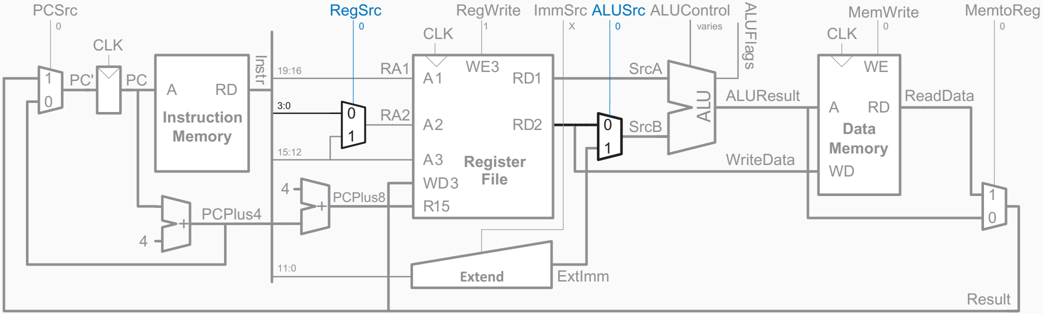


Figure 7.11 Datapath enhancements for data-processing instructions with register addressing

Execução da instrução B

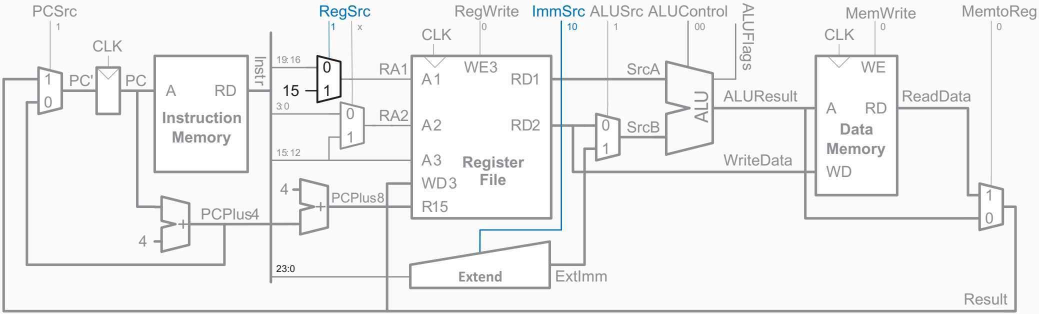


Figure 7.12 Datapath enhancements for B instruction

Resumo das opções de imediatos

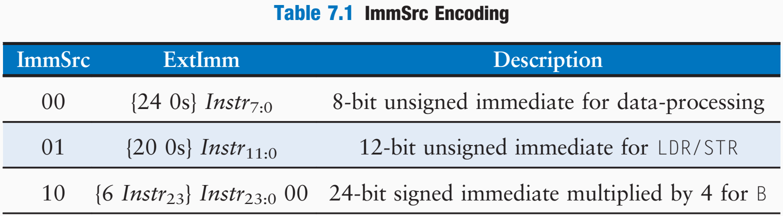


Table 7.1 ImmSrc Encoding

ImmSrc	ExtImm	Description
00	{24 0s} $Instr_{7:0}$	8-bit unsigned immediate for data-processing
01	{20 0s} $Instr_{11:0}$	12-bit unsigned immediate for `LDR`/`STR`
10	{6 $Instr_{23}$} $Instr_{23:0}$ 00	24-bit signed immediate multiplied by 4 for `B`

Processador monociclo completo

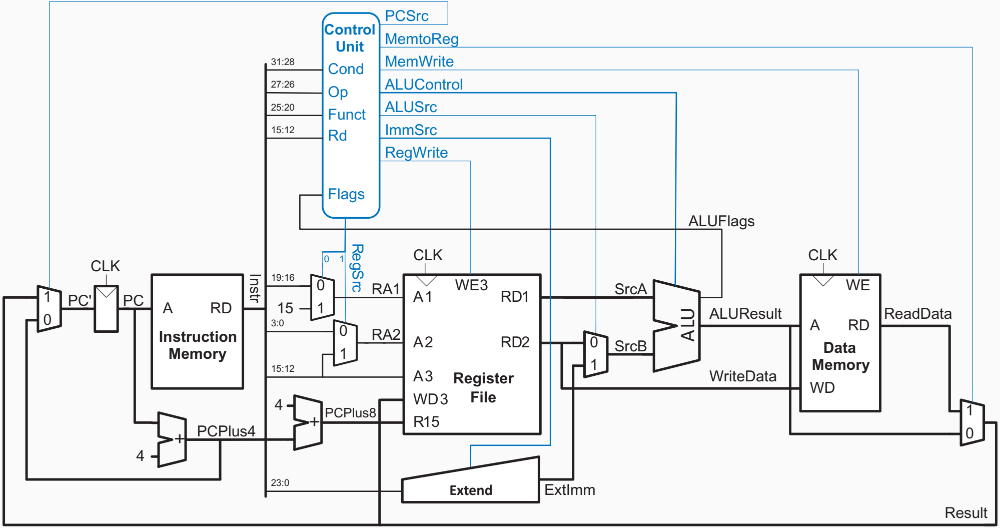


Figure 7.13 Complete single-cycle processor

Unidade de controle

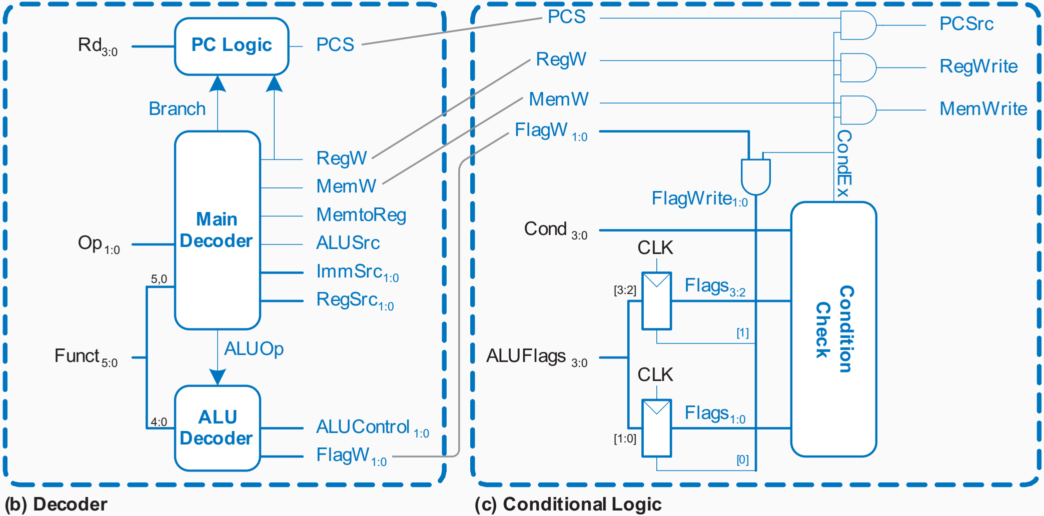


Figure 7.14 Single-cycle control unit

Decodificador principal

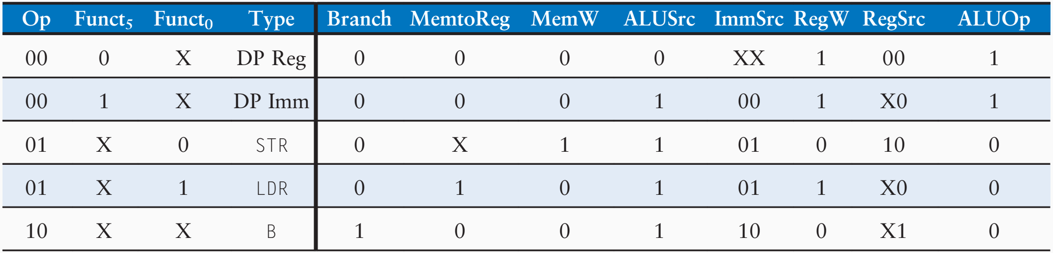
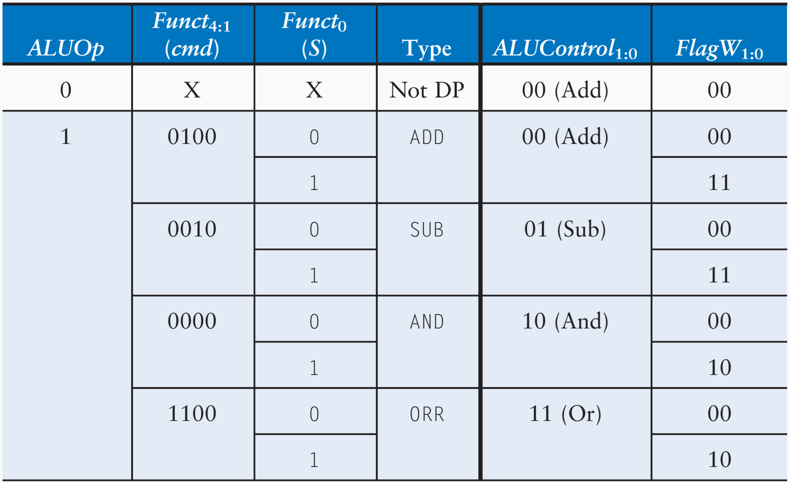


Table 7.2 **Main Decoder truth table**

Op	$Funct_5$	$Funct_0$	Type	Branch	MemtoReg	MemW	ALUSrc	ImmSrc	RegW	RegSrc	ALUOp
00	0	X	DP Reg	0	0	0	0	XX	1	00	1
00	1	X	DP Imm	0	0	0	1	00	1	X0	1
01	X	0	STR	0	X	1	1	01	0	10	0
01	X	1	LDR	0	1	0	1	01	1	X0	0
10	X	X	B	1	0	0	1	10	0	X1	0

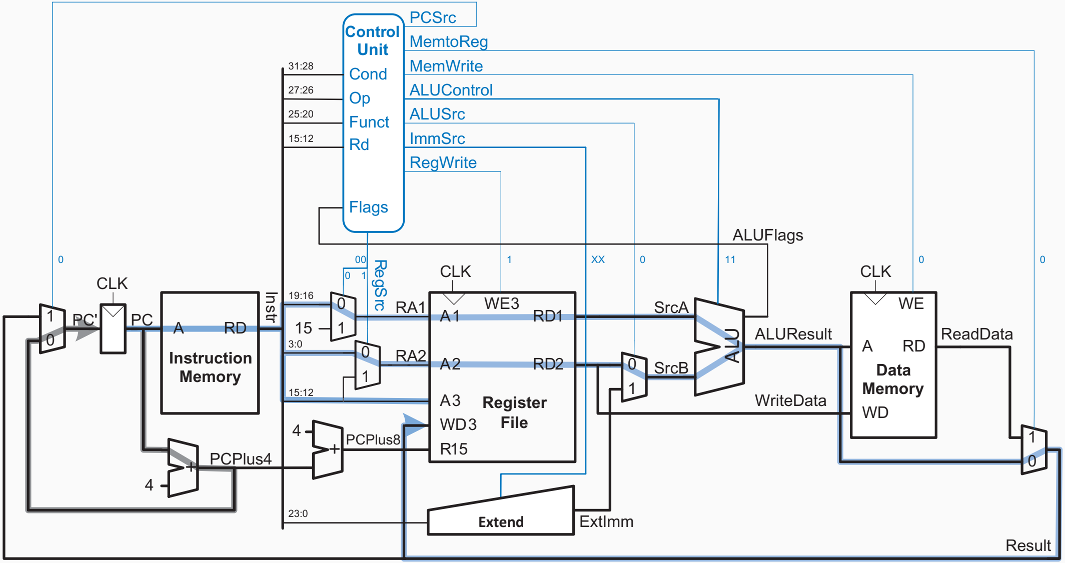
Decodificador da ULA

Table 7.3 ALU Decoder truth table

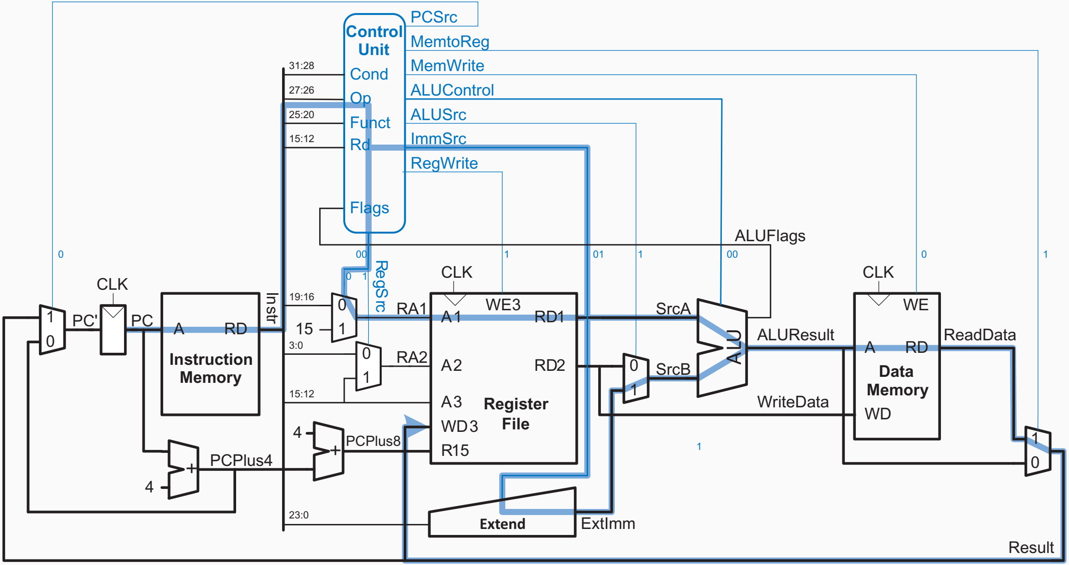


$ALUOp$	$Funct_{4:1}$ (cmd)	$Funct_0$ (S)	Type	$ALUControl_{1:0}$	$FlagW_{1:0}$
0	X	X	Not DP	00 (Add)	00
1	0100	0	ADD	00 (Add)	00
		1			11
	0010	0	SUB	01 (Sub)	00
		1			11
	0000	0	AND	10 (And)	00
		1			10
	1100	0	ORR	11 (Or)	00
		1			10

Dados e controles na instrução ORR



Caminho crítico na instrução LDR



Processador multiciclo

Execução da instrução LDR

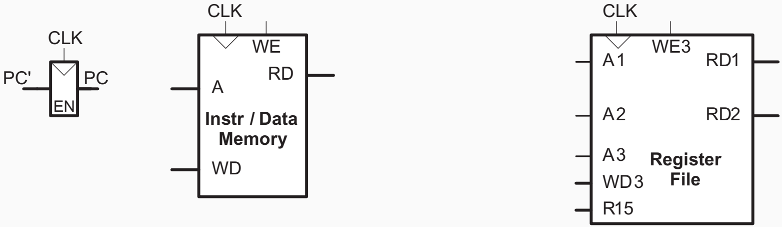


Figure 7.19 **State elements with unified instruction/data memory**

Execução da instrução LDR

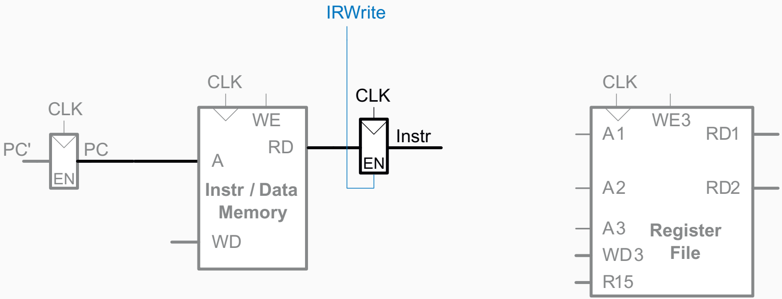


Figure 7.20 **Fetch instruction from memory**

Execução da instrução LDR

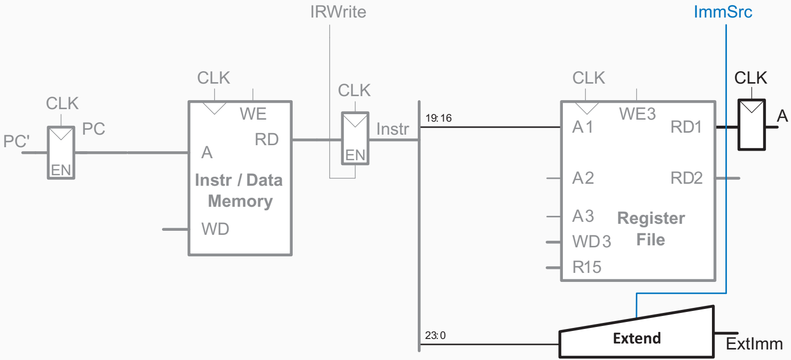


Figure 7.21 Read one source from register file and extend the second source from the immediate field

Execução da instrução LDR

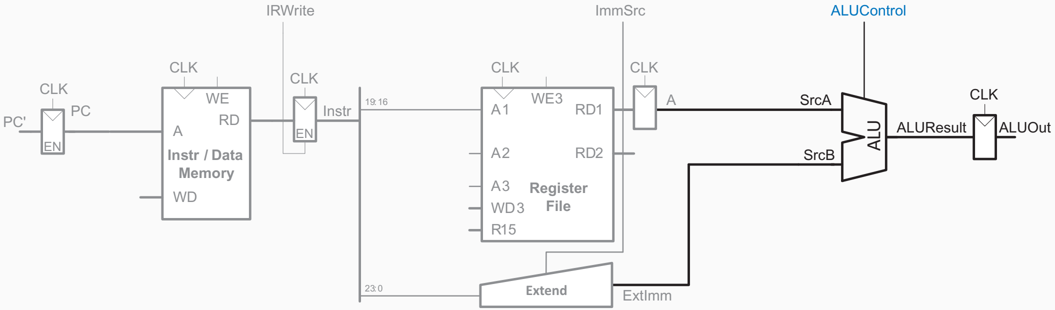


Figure 7.22 **Add base address to offset**

Execução da instrução LDR

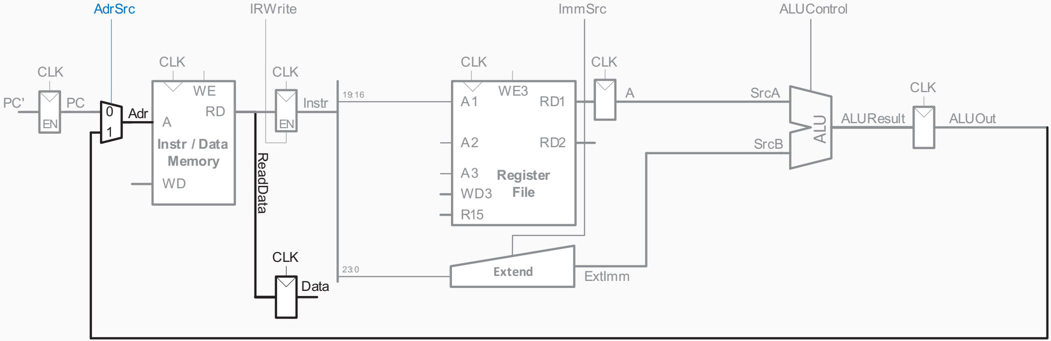


Figure 7.23 **Load data from memory**

Execução da instrução LDR

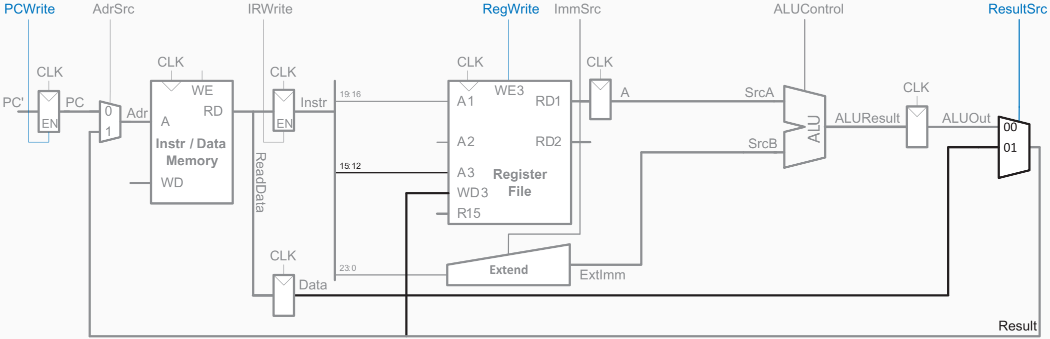


Figure 7.24 Write data back to register file

Execução da instrução LDR

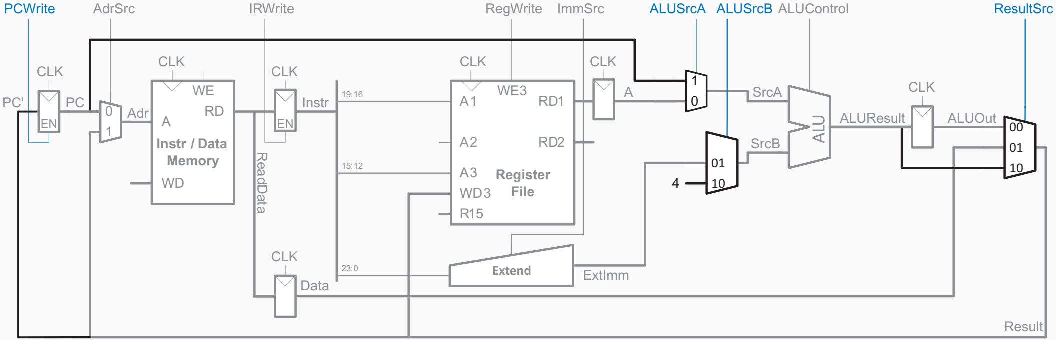


Figure 7.25 **Increment PC by 4**

Execução da instrução LDR

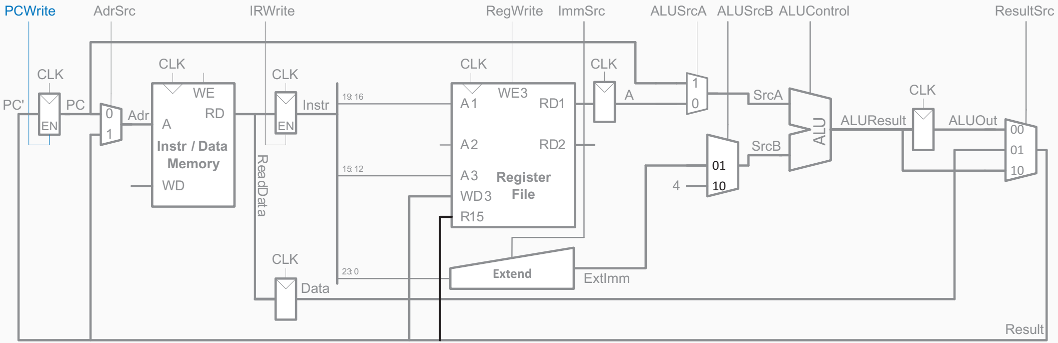


Figure 7.26 **Handle R15 reads and writes**

Execução da instrução STR

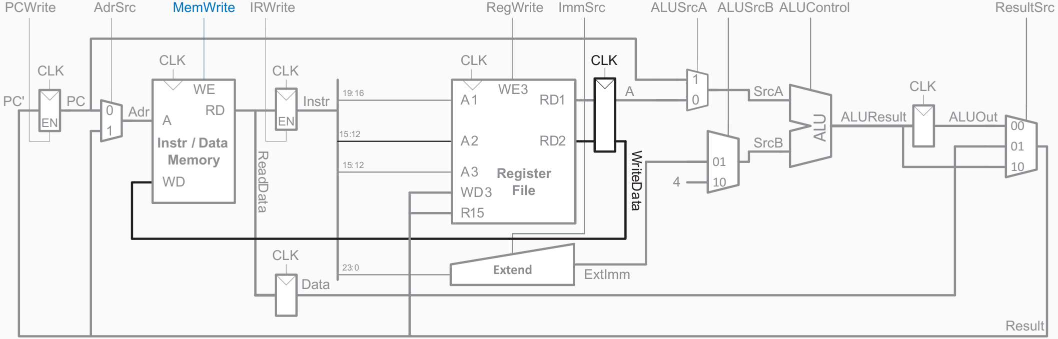


Figure 7.27 Enhanced datapath for STR instruction

Processador multiciclo completo

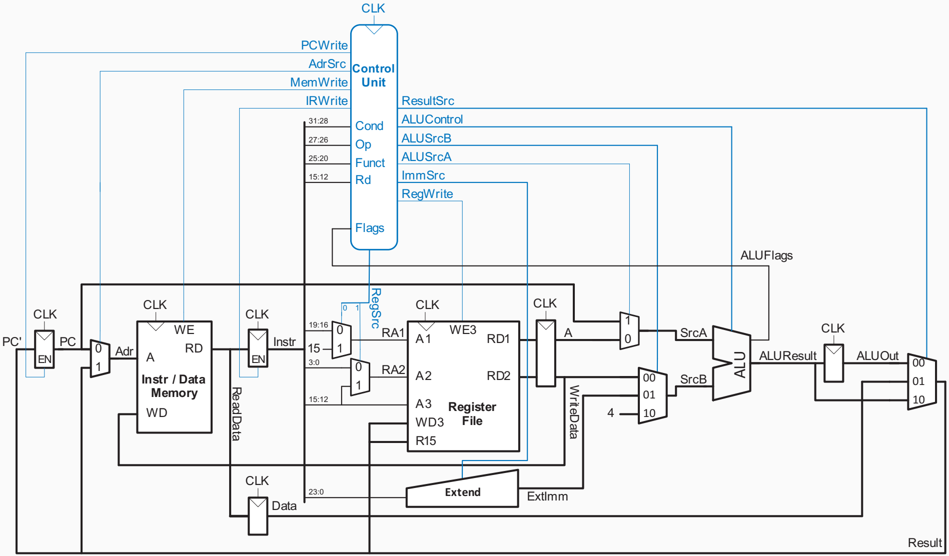
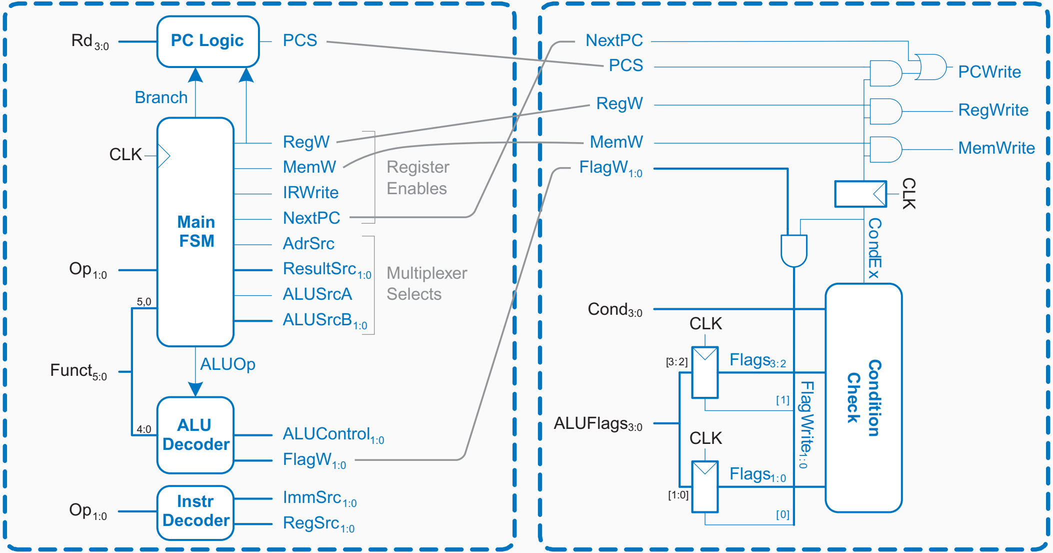


Figure 7.30 Complete multicycle processor

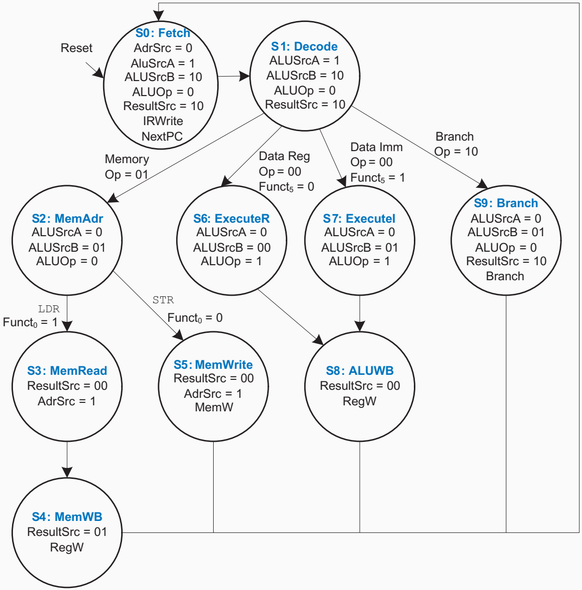
Unidade de controle



(b) Decoder

(c) Conditional Logic

Maquina de Estados



State	**Datapath µOp**
Fetch	Instr ←Mem[PC]; PC ← PC+4
Decode	ALUOut ← PC +4
MemAdr	ALUOut ← Rn + Imm
MemRead	Data ← Mem[ALUOut]
MemWB	Rd ← Data
MemWrite	Mem[ALUOut] ← Rd
ExecuteR	ALUOut ← Rn op Rm
ExecuteI	ALUOut ← Rn op Imm
ALUWB	Rd ← ALUOut
Branch	PC ← R15 + offset

Figure 7.41 Complete multicycle control FSM

Processador pipeline

Processador pipeline

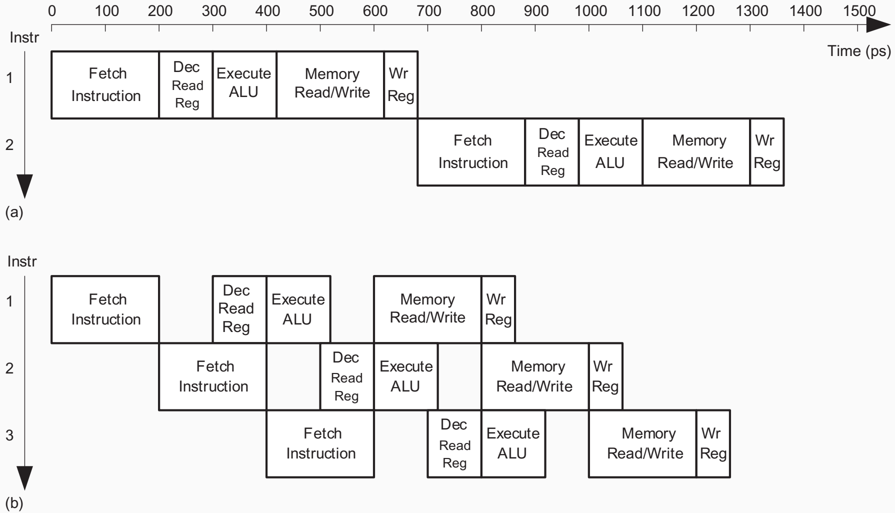
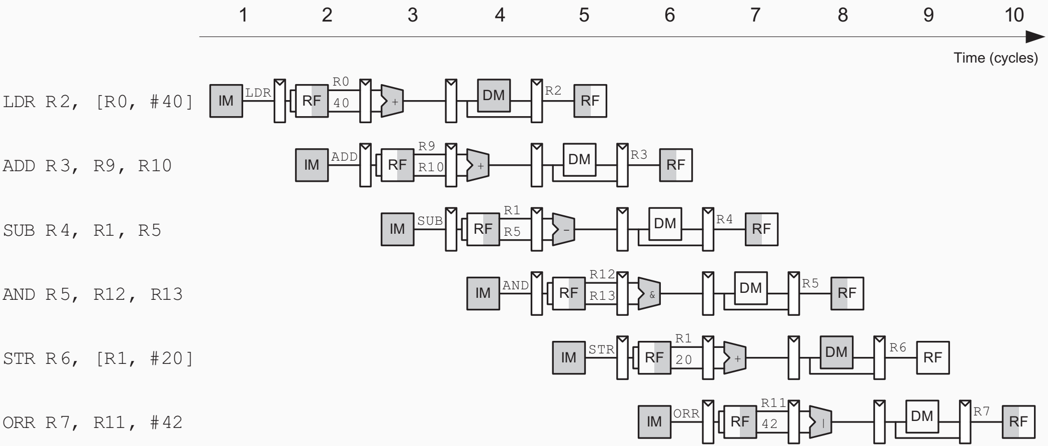


Figure 7.42 **Timing diagrams: (a) single-cycle processor and (b) pipelined processor**

Processador pipeline



```
LDR R2, [R0, #40]
ADD R3, R9, R10
SUB R4, R1, R5
AND R5, R12, R13
STR R6, [R1, #20]
ORR R7, R11, #42
```

Figure 7.43 Abstract view of pipeline in operation

Processador pipeline

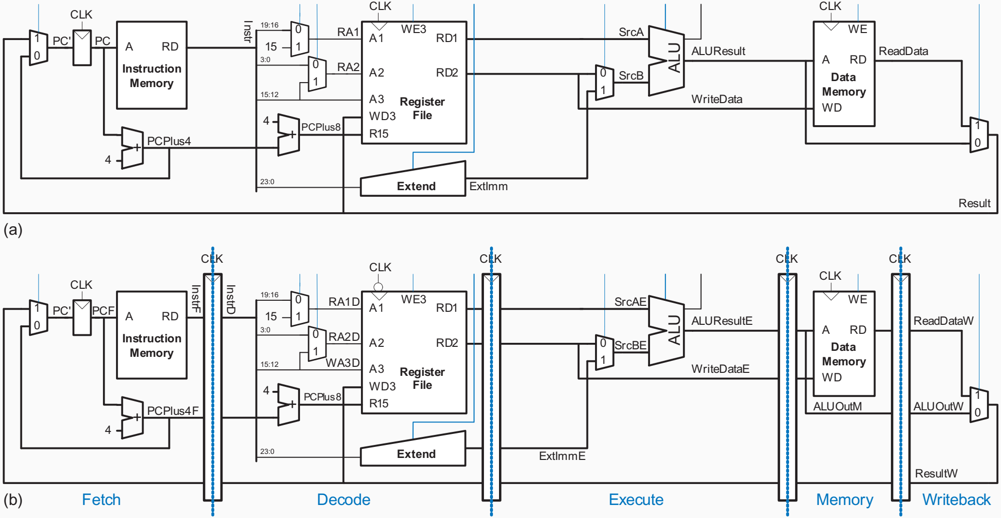


Figure 7.44 **Datapaths: (a) single-cycle and (b) pipelined**

Referências

Para saber mais e praticar...

- https://www.arm.com/resources
- https://salmanarif.bitbucket.io/visual/
- https://cpulator.01xz.net/?sys=arm
- https://azm.azerialabs.com/
- https://www.edaplayground.com/x/vcGc
- https://booksite.elsevier.com/9780128000564/lab_materials.php

Bibliografia

David Harris and Sarah Harris.
Digital Design and Computer Architecture: ARM® Edition.
Morgan Kaufmann, 2016.