

# Computer Architecture Project - Phase 1

## Team 1

Ahmad Khaled	S01 BN 03
Zeinab Rabie	S01 BN 26
Sara Maher	S01 BN 28
Abdelrahman Mohamed	S02 BN 03

# 1 Instruction Set Architecture

The structure for the IR for two-operand instructions is given in Figure 1. The structure for one-operand instructions is given in Figure 2. The structure for branching instructions is given in Figure 3. The structure for the rest of the instructions is given in Figure 4.

<b>Direct</b>	Register	Auto-increment	Auto-decrement	Indexed
<i>Code</i>	000	001	010	011
<b>Indirect</b>	Register	Auto-increment	Auto-decrement	Indexed
<i>Code</i>	100	101	110	111

Table 1: Addressing Mode Codes

	$R_0$	$R_1$	$R_2$	$R_3$
<i>Code</i>	000	001	010	011
	$R_6$	$R_5$	$R_7$	$R_8$
<i>Code</i>	100	101	110	111

Table 2: Register Addressing Codes

Instruction	MOV	ADD	ADC	SUB	SBC	AND	OR	XNOR	CMP
OP Code	1111	1110	1101	1100	1011	1010	1001	1000	0111

Table 3: Two-Operand Instruction Codes

Figure 1: IR Structure For Two-Operand Instructions

15	12	11	9	8	6	5	3	2	0
Instruction		Source Address		Source Register		Destination Address		Destination Register	

**Instruction** specifies the instruction to execute. Possible codes given in Table 3.

**Source Address** specifies the addressing mode to choose for the source. Possible codes given in Table 1.

**Source Register** specifies the source register. Possible codes given in Table 2.

**Destination Address** specifies the addressing mode to choose for the destination. Possible codes given in Table 1.

**Destination Register** specifies the destination register. Possible codes given in Table 2.

Instruction	INC	DEC	CLR	INV	LSR	ROR	RRC	ASR	LSL	ROL	RLC
OP Code	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010

Table 4: Single Operand Instruction Codes

Figure 2: IR Structure For Single-Operand Instructions

1512				118				76		53				20			
0110				Instruction				00		Operand Address				Operand Register			

**Instruction** specifies the instruction to execute. Possible codes given in Table 4.

**Operand Address** specifies the addressing mode to choose for the operand. Possible codes given in Table 1.

**Operand Register** specifies the operand register. Possible codes given in Table 2.

Instruction	BR	BEQ	BNE	BLO	BLS	BHI	BHS
OP Code	000	001	010	011	100	101	110

Table 5: Branching Instruction Codes

Figure 3: IR Structure For Branching Instructions

15	12	11	9	8	0
0	1	0	1	Instruction	Operand

**Instruction** specifies the instruction to execute. Possible codes given in Table 5.

**Operand** specifies the branching offset.

Instruction	JSR	RTS	HITR	IRET	HLT	NOP
OP Code	0100	0011	0011	0010	0001	0000

Table 6: Miscellaneous Instruction Codes

Figure 4: IR Structure For Miscellaneous Instructions

15	12	11	10	0
Instruction		Extra	Operand	

**Instruction** specifies the instruction to execute (except for differentiating RTS and HITR, done using Extra bit). Possible codes given in Table 6.

**Extra** useful for JSR and HITR only. specifies whether the Operand is given directly or using a register/addressing mode for JSR. Indicates HITR when instruction is 0011 and Extra= 1, indicates RTS if instruction is 0011 and Extra= 0.

**Operand** useful for JSR only. specifies the memory location to jump to (if Extra= 0) or register/addressing mode otherwise.

## 2 Bus System Schematic

Figure 5 includes the schematic for the microprocessor design. The register file contains both the general purpose registers as well as the special purpose ones. Note: according to the schematic, both TMP1 and TMP2 are connected to the ALU but by default  $B$  is connected to TMP1 (and converted to TMP2 by a special signal if needed).

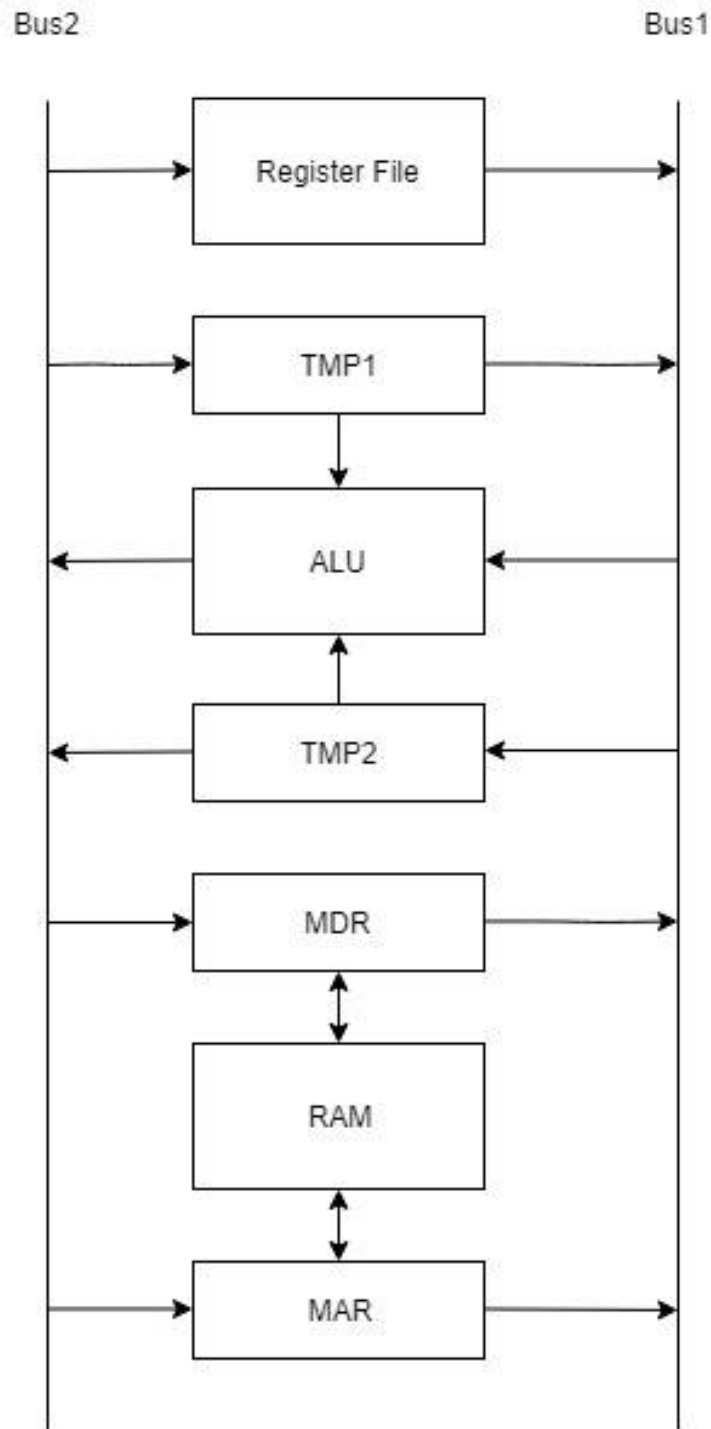


Figure 5: Bus System Schematic

## 3 Micro-Instructions

### 3.1 Control Signals

We analyze the Fetch / Execution phase each on its own and mention the MA / Clock Cycles that are associated only with this phase. See the next subsection for the total analysis.

1. **Fetch and Decode.** MA = 1, Clock Cycles = 3.

- (a)  $PC_{out}, F = A, MAR_{in}, TMP1_{in}, Rd$
- (b)  $F = A + 1, PC_{in}, WMFC$
- (c)  $MDR_{out}, F = A, IR_{in}$

2. **Fetch Op.** Please note that if two operands are desired, the result of the source operand (outed in the last line) is output to TMP2 register instead of TMP1 at the end, then the destination is fetched and placed in TMP1.

- (a)  $Op = R_i$ . MA = 0, Clock Cycles = 1.
  - i.  $R_{iout}, F = A, TMP1_{in}$
- (b)  $Op = (R_i)+$ . MA = 1, Clock Cycles = 3.
  - i.  $R_{iout}, F = A, MAR_{in}, TMP1_{in}, Rd$
  - ii.  $F = B + 1, R_{iin}, WMFC$
  - iii.  $MDR_{out}, F = A, TMP1_{in}$
- (c)  $Op = -(R_i)$ . MA = 1, Clock Cycles = 3.
  - i.  $R_{iout}, F = A - 1, MAR_{in}, TMP1_{in}, Rd$
  - ii.  $F = B, R_{iin}, WMFC$
  - iii.  $MDR_{out}, F = A, TMP1_{in}$
- (d)  $Op = X(R_i)$ . MA = 2, Clock Cycles = 4.
  - i.  $PC_{out}, F = A, MAR_{in}, Rd, TMP1_{in}$
  - ii.  $TMP1_{out}, F = A + 1, PC_{in}, WMFC$
  - iii.  $MDR_{out}, F = A + B + 1, MAR_{in}, Rd, WMFC$
  - iv.  $MDR_{out}, F = A, TMP1_{in}$
- (e)  $Op = @R_i$ . MA = 1, Clock Cycles = 2.
  - i.  $R_{iout}, F = A, MAR_{in}, Rd, WMFC$
  - ii.  $MDR_{out}, F = A, TMP1_{in}$
- (f)  $Op = @(R_i)+$ . MA = 2, Clock Cycles = 4.
  - i.  $R_{iout}, F = A, MAR_{in}, TMP1_{in}, Rd$
  - ii.  $F = B + 1, R_{iin}, WMFC$
  - iii.  $MDR_{out}, F = A, MAR_{in}, Rd, WMFC$
  - iv.  $MDR_{out}, F = A, TMP1_{in}$
- (g)  $Op = @-(R_i)$ . MA = 2, Clock Cycles = 4.
  - i.  $R_{iout}, F = A - 1, MAR_{in}, TMP1_{in}, Rd$

- ii.  $F = B, R_{in}, \text{WMFC}$
  - iii.  $\text{MDR}_{out}, F = A, \text{MAR}_{in}, \text{Rd}, \text{WMFC}$
  - iv.  $\text{MDR}_{out}, F = A, \text{TMP1}_{in}$
- (h)  $\text{Op} = @X(R_i)$ .  $\text{MA} = 3$ , Clock Cycles = 5.
- i.  $\text{PC}_{out}, F = A, \text{MAR}_{in}, \text{Rd}, \text{TMP1}_{in}$
  - ii.  $\text{TMP1}_{out}, F = A + 1, \text{PC}_{in}, \text{WMFC}$
  - iii.  $\text{MDR}_{out}, F = A + B + 1, \text{MAR}_{in}, \text{Rd}, \text{WMFC}$
  - iv.  $\text{MDR}_{out}, F = A, \text{MAR}_{in}, \text{Rd}, \text{WMFC}$
  - v.  $\text{MDR}_{out}, F = A, \text{TMP1}_{in}$
3. **Two-Operand Execution Instructions.**  $\text{MA} = 0$  if Dst is a register, 1 otherwise. Clock Cycles = 1.  
 We assume that the Source operand is in TMP2 (connected to the ALU as  $B$ ) and the Destination operand is in TMP1.  $\text{Dst}_{in}$  is replaced by  $\text{MDR}_{in}$ , Write if Dst is a memory location and is replaced by  $R_{in}$  if Dst is a register.
- (a) MOV Src, Dst
    - i.  $\text{TMP2}_{out}, F = A, \text{Dst}_{in}$
  - (b) ADD Src, Dst
    - i.  $\text{TMP2}_{out}, F = A + B, \text{Dst}_{in}$
  - (c) ADC Src, Dst
    - i.  $\text{TMP2}_{out}, F = A + B + C, \text{Dst}_{in}$
  - (d) SUB Src, Dst
    - i.  $\text{TMP2}_{out}, F = B - A, \text{Dst}_{in}$
  - (e) SBC Src, Dst
    - i.  $\text{TMP2}_{out}, F = B - A - C, \text{Dst}_{in}$
  - (f) AND Src, Dst
    - i.  $\text{TMP2}_{out}, F = A \text{ AND } B, \text{Dst}_{in}$
  - (g) OR Src, Dst
    - i.  $\text{TMP2}_{out}, F = A \text{ OR } B, \text{Dst}_{in}$
  - (h) XNOR Src, Dst
    - i.  $\text{TMP2}_{out}, F = A \text{ XNOR } B, \text{Dst}_{in}$
  - (i) CMP Src, Dst
    - i.  $\text{TMP2}_{out}, F = B - A$
4. **Single-Operand Instructions.**  $\text{MA} = 0$  if Dst is a register, 1 otherwise. Clock Cycles = 1.  
 We assume that the destination operand is in TMP1.  $\text{Dst}_{in}$  is replaced by  $\text{MDR}_{in}$ , Write if Dst is a memory location and is replaced by  $R_{in}$  if Dst is a register.
- (a) INC Dst
    - i.  $F = B + 1, \text{Dst}_{in}$

- (b) DEC Dst
  - i.  $F = B - 1, \text{Dst}_{\text{in}}$
- (c) CLR Dst
  - i.  $F = 0, \text{Dst}_{\text{in}}$
- (d) INV Dst
  - i.  $F = \text{Not}(B), \text{Dst}_{\text{in}}$
- (e) LSR Dst
  - i.  $F = \text{LSR}(B), \text{Dst}_{\text{in}}$
- (f) ROR Dst
  - i.  $F = \text{ROR}(B), \text{Dst}_{\text{in}}$
- (g) RRC Dst
  - i.  $F = \text{RRC}(B), \text{Dst}_{\text{in}}$
- (h) ASR Dst
  - i.  $F = \text{ASR}(B), \text{Dst}_{\text{in}}$
- (i) LSL Dst
  - i.  $F = \text{LSL}(B), \text{Dst}_{\text{in}}$
- (j) ROL Dst
  - i.  $F = \text{ROL}(B), \text{Dst}_{\text{in}}$
- (k) RLC Dst
  - i.  $F = \text{RLC}(B), \text{Dst}_{\text{in}}$

5. **No Operand Instructions.** MA = 0, Clock Cycles = 1. Assume the existence of HLT/NOP control signals.

- (a) HLT.
  - i. HLT
- (b) NOP.
  - i. NOP

6. **Branching Instructions.** MA = 0, Clock Cycles = 2.

- (a) BR
  - i.  $\text{PC}_{\text{out}}, F = A, \text{TMP1}_{\text{in}}$
  - ii. Offset Part of  $\text{IR}_{\text{out}}, F = A + B, \text{PC}_{\text{in}}$
- (b) BEQ
  - i.  $\text{PC}_{\text{out}}, F = A, \text{TMP1}_{\text{in}}, \text{If } Z = 1 \text{ then End}$
  - ii. Offset Part of  $\text{IR}_{\text{out}}, F = A + B, \text{PC}_{\text{in}}$
- (c) BNEQ
  - i.  $\text{PC}_{\text{out}}, F = A, \text{TMP1}_{\text{in}}, \text{If } Z = 0 \text{ then End}$



- ii. Offset Part of  $IR_{out}$ ,  $F = A + B$ ,  $PC_{in}$
- (d) BLO
  - i.  $PC_{out}$ ,  $F = A$ ,  $TMP1_{in}$ , If  $C = 0$  then End
  - ii. Offset Part of  $IR_{out}$ ,  $F = A + B$ ,  $PC_{in}$
- (e) BLS
  - i.  $PC_{out}$ ,  $F = A$ ,  $TMP1_{in}$ , If  $C = 0$  or  $Z = 1$  then End
  - ii. Offset Part of  $IR_{out}$ ,  $F = A + B$ ,  $PC_{in}$
- (f) BHI
  - i.  $PC_{out}$ ,  $F = A$ ,  $TMP1_{in}$ , If  $C = 1$  then End
  - ii. Offset Part of  $IR_{out}$ ,  $F = A + B$ ,  $PC_{in}$
- (g) BHS
  - i.  $PC_{out}$ ,  $F = A$ ,  $TMP1_{in}$ , If  $C = 1$  or  $Z = 1$  then End
  - ii. Offset Part of  $IR_{out}$ ,  $F = A + B$ ,  $PC_{in}$

## 7. Jump Instructions.

- (a) JSR. MA = 3, Clock Cycles = 7.
  - i.  $PC_{out}$ ,  $F = A$ ,  $MAR_{in}$ ,  $TMP1_{in}$ , Rd
  - ii.  $F = B + 1$ ,  $PC_{in}$ , WMFC
  - iii.  $MDR_{out}$ ,  $F = A$ ,  $TMP2_{in}$
  - iv.  $SP_{out}$ ,  $F = A - 1$ ,  $TMP1_{in}$
  - v.  $F = B$ ,  $SP_{in}$ ,  $MAR_{in}$
  - vi.  $PC_{out}$ ,  $F = A$ ,  $MDR_{in}$ , Write, WMFC
  - vii.  $TMP2_{out}$ ,  $F = A$ ,  $PC_{in}$
- (b) RTS. MA = 2, Clock Cycles = 3.
  - i.  $SP_{out}$ ,  $F = A$ ,  $MAR_{in}$ ,  $TMP2_{in}$ , Rd
  - ii.  $TMP2_{out}$ ,  $F = A + 1$ ,  $SP_{in}$ , WMFC
  - iii.  $MDR_{out}$ ,  $F = A$ ,  $PC_{in}$
- (c) HIR. MA = 3, Clock Cycles = 6.  
Assuming the address to go to is stored in the IR.
  - i.  $SP_{out}$ ,  $F = A - 1$ ,  $TMP1_{in}$
  - ii.  $TMP1_{in}$ ,  $F = A$ ,  $MAR_{in}$ ,  $SP_{in}$
  - iii.  $FLAG_{out}$ ,  $F = A$ ,  $MDR_{in}$ , Write, WMFC
  - iv.  $F = B - 1$ ,  $MAR_{in}$ ,  $SP_{in}$
  - v.  $PC_{out}$ ,  $F = A$ ,  $MDR_{in}$ , Write, WMFC
  - vi. Address Part of  $IR_{out}$ ,  $F = A$ ,  $PC_{in}$
- (d) IRET. MA = 3, Clock Cycles = 6.
  - i.  $SP_{out}$ ,  $F = A$ ,  $MAR_{in}$ ,  $TMP2_{in}$ , Read
  - ii.  $TMP2_{out}$ ,  $F = A + 1$ ,  $SP_{in}$ ,  $TMP1_{in}$ , WMFC
  - iii.  $MDR_{out}$ ,  $F = A$ ,  $PC_{in}$

- iv.  $\text{TMP1}_{\text{out}}, \text{MAR}_{\text{in}}, \text{Read}$
- v.  $F = B + 1, \text{SP}_{\text{in}}, \text{WMFC}$
- vi.  $\text{MDR}_{\text{out}}, F = A, \text{FLAG}_{\text{in}}$

### 3.2 Memory and Cycle Analysis

The average for all instructions is given in Table 7. The MA / Clock Cycles in this subsection include the cycles and accesses required for fetching and decoding as well as fetching operands when necessary.

Instruction	Two-Op (9)	Single-Op (11)	Zero-Op (2)	Branching (7)	Average (Excl Jumps)
Average MA	4.875	3.375	1	1	<b>3.1034</b>
Average CC	10.5	7.25	4	5	<b>7.4913</b>

Table 7: Average MA / Clock Cycles for design. The number between braces in the instruction name is the count of instructions belonging to that category.

The detailed analyses for two-operand instructions according to source / destination addressing modes are given in Table 8 for MA and in Table 9 for Clock Cycles. The detailed analysis for single-operand instructions according to operand address is given in Table 10. The Jumps are given in Table 11.

Src Dst	$R_i$	$-(R_i) / (R_i)+ / @R_i$	$X(R_i) / @(R_i)+ / @-(R_i)$	$@X(R_i)$
$R_j$	1	2	3	4
$-(R_j)$ $(R_j)+$ $@R_j$	3	4	5	6
$X(R_j)$ $ @(R_j)+$ $@-(R_j)$	4	5	6	7
$@X(R_j)$	5	6	7	8

Table 8: Memory-Access Two-Operand Instruction Analysis.

Src Dst	$R_i$	$@R_i$	$-(R_i) / (R_i)+$	$X(R_i) / @(R_i)+ / @-(R_i)$	$@X(R_i)$
$R_j$	6	7	8	9	10
$@R_i$	7	8	9	10	11
$-(R_j)$ $(R_j)+$	8	9	10	11	12
$X(R_j)$ $ @(R_j)+$ $@-(R_j)$	9	10	11	12	13
$@X(R_j)$	10	11	12	13	14

Table 9: Clock Cycles Two-Operand Instruction Analysis.

Op	$R_i$	$@R_i$	$-(R_i) / (R_i)+$	$X(R_i) / @ (R_i)+ / @ - (R_i)$	$@X(R_i)$
MA	1	3	3	4	5
Clock Cycles	5	6	7	8	9

Table 10: MA and Clock Cycle Instruction Analysis for Single Operand Instructions

Instruction	JSR	RTS	HITR	IRET	Average
MA	4	3	3	4	3.5
Clock Cycles	10	6	6	9	7.75

Table 11: MA and Clock Cycle Instruction Analysis for Jump Instructions