Architecture Project

Simple PDP-11

Team Members	Sec	Bn
-Evram Youssef	1	9
-Remonda Talaat	1	20
-Mohamed Shawky	2	16
-Mahmoud Othman	2	21

Submitted to: TA.Ali el-Seddiq

Under supervision of: Dr. Hwaida

Memory Access Analysis

Group 1: Two Operands Regular Instructions (ADD, ADC, SUB, SUBC, AND, OR, XNOR):

Addressing modes are (RGS: register, IND_RGS: indirect register)

SRC	DST	MA
RGS	RGS	1
RGS	IND_RGS	3
RGS	Autoincremet	3
RGS	IND_autoincremetn	4
RGS	Autodecrement	3
RGS	IND_Autodec	4
RGS	Indexed	4
RGS	Ind_Indexed	5
INC	RGS	2
INC	IND_RGS	4
INC	Autoincremet	4
INC	IND_autoincremetn	5
INC	Autodecrement	4
INC	IND_Autodec	5
INC	Indexed	5
INC	Ind_Indexed	6
DEC	RGS	2
DEC	IND_RGS	4
DEC	Autoincremet	4
DEC	IND_autoincremetn	5
DEC	Autodecrement	4
DEC	IND_Autodec	5
DEC	Indexed	5
DEC	Ind_Indexed	6
INDEX	RGS	3
INDEX	IND_RGS	5
INDEX	Autoincremet	5
INDEX	IND_autoincremetn	6
INDEX	Autodecrement	5
INDEX	IND_Autodec	6
INDEX	Indexed	6
INDEX	Ind_Indexed	7

SRC	DST	MA
IND_RGS	RGS	2
IND_RGS	IND_RGS	4
IND_RGS	Autoincremet	4
IND_RGS	IND_autoincremetn	5
IND_RGS	Autodecrement	4
IND_RGS	IND_Autodec	5
IND_RGS	Indexed	5
IND_RGS	Ind_Indexed	6
IND_INC	RGS	3
IND_INC	IND_RGS	5
IND_INC	Autoincremet	5
IND_INC	IND_autoincremetn	6
IND_INC	Autodecrement	5
IND_INC	IND_Autodec	6
IND_INC	Indexed	6
IND_INC	Ind_Indexed	7
IND_DEC	RGS	3
IND_DEC	IND_RGS	5
IND_DEC	Autoincremet	5
IND_DEC	IND_autoincremetn	6
IND_DEC	Autodecrement	5
IND_DEC	IND_Autodec	6
IND_DEC	Indexed	6
IND_DEC	Ind_Indexed	7
IND_INDEX	RGS	4
IND_INDEX	IND_RGS	6
IND_ INDEX	Autoincremet	6
IND_ INDEX	IND_autoincremetn	7
IND_ INDEX	Autodecrement	6
IND_ INDEX	IND_Autodec	7
IND_ INDEX	Indexed	7
IND_ INDEX	Ind_Indexed	8

Group 2: Two Operands Special Instructions (MOV, CMP):

As MOV doesn't require the final data of the destination, just the address, and CMP doesn't store the final value of the operation.

So, the second table would differ from the previous one, as MA would be less by 1, except when RGS is at the destination.

SRC	DST	MA
RGS	RGS	**1**
RGS	IND_RGS	2
RGS	Autoincremet	2
RGS	IND_autoincremetn	3
RGS	Autodecrement	2
RGS	IND_Autodec	3
RGS	Indexed	3
RGS	Ind_Indexed	4
INC	RGS	**2**
INC	IND_RGS	3
INC	Autoincremet	3
INC	IND_autoincremetn	4
INC	Autodecrement	3
INC	IND_Autodec	4
INC	Indexed	4
INC	Ind_Indexed	5
DEC	RGS	**2**
DEC	IND_RGS	3
DEC	Autoincremet	3
DEC	IND_autoincremetn	4
DEC	Autodecrement	3
DEC	IND_Autodec	4
DEC	Indexed	4
DEC	Ind_Indexed	5
INDEX	RGS	**3**
INDEX	IND_RGS	4
INDEX	Autoincremet	4
INDEX	IND_autoincremetn	5
INDEX	Autodecrement	4
INDEX	IND_Autodec	5
INDEX	Indexed	5
INDEX	Ind_Indexed	6

SRC	DST	MA
IND_RGS	RGS	**2**
IND_RGS	IND_RGS	3
IND_RGS	Autoincremet	3
IND_RGS	IND_autoincremetn	4
IND_RGS	Autodecrement	3
IND_RGS	IND_Autodec	4
IND_RGS	Indexed	4
IND_RGS	Ind_Indexed	5
IND_INC	RGS	**3**
IND_INC	IND_RGS	4
IND_INC	Autoincremet	4
IND_INC	IND_autoincremetn	5
IND_INC	Autodecrement	4
IND_INC	IND_Autodec	5
IND_INC	Indexed	5
IND_INC	Ind_Indexed	6
IND_DEC	RGS	**3**
IND_DEC	IND_RGS	4
IND_DEC	Autoincremet	4
IND_DEC	IND_autoincremetn	5
IND_DEC	Autodecrement	4
IND_DEC	IND_Autodec	5
IND_DEC	Indexed	5
IND_DEC	Ind_Indexed	6
IND_INDEX	RGS	**4**
IND_INDEX	IND_RGS	5
IND_ INDEX	Autoincremet	5
IND_ INDEX	IND_autoincremetn	6
IND_ INDEX	Autodecrement	5
IND_ INDEX	IND_Autodec	6
IND_ INDEX	Indexed	6
IND_ INDEX	Ind_Indexed	7

Group3: One Operand Instructions (INC, DEC, CLR, INV, LSR, ROR, RRC, ASR, LSL, ROL, RLC):

Destination	**MA**
Register	**1**
Indirect Register	3
Autoincremet	3
Indirect Autoincremetn	4
Autodecrement	3
Indirect Autodecrement	4
Indexed	4
Indirect Indexed	5

Group 4: Jumpers and Stackers (JSR, RTS, ITR, IRET):

Operation	Memory Access
JSR	**3**
RTS	**2**
ITR	**3**
IRET	**3**

Group 5: Branches, HLT and NO Operation:

Memory Access = 1 for fetching any of these instructions, no memory accessing is required for any of them.

Architecture:

#of General Purpose Registers = 8 (including stack register and program counter)

#of added TEMP registers = 3 (SOURCE, Y, Z)

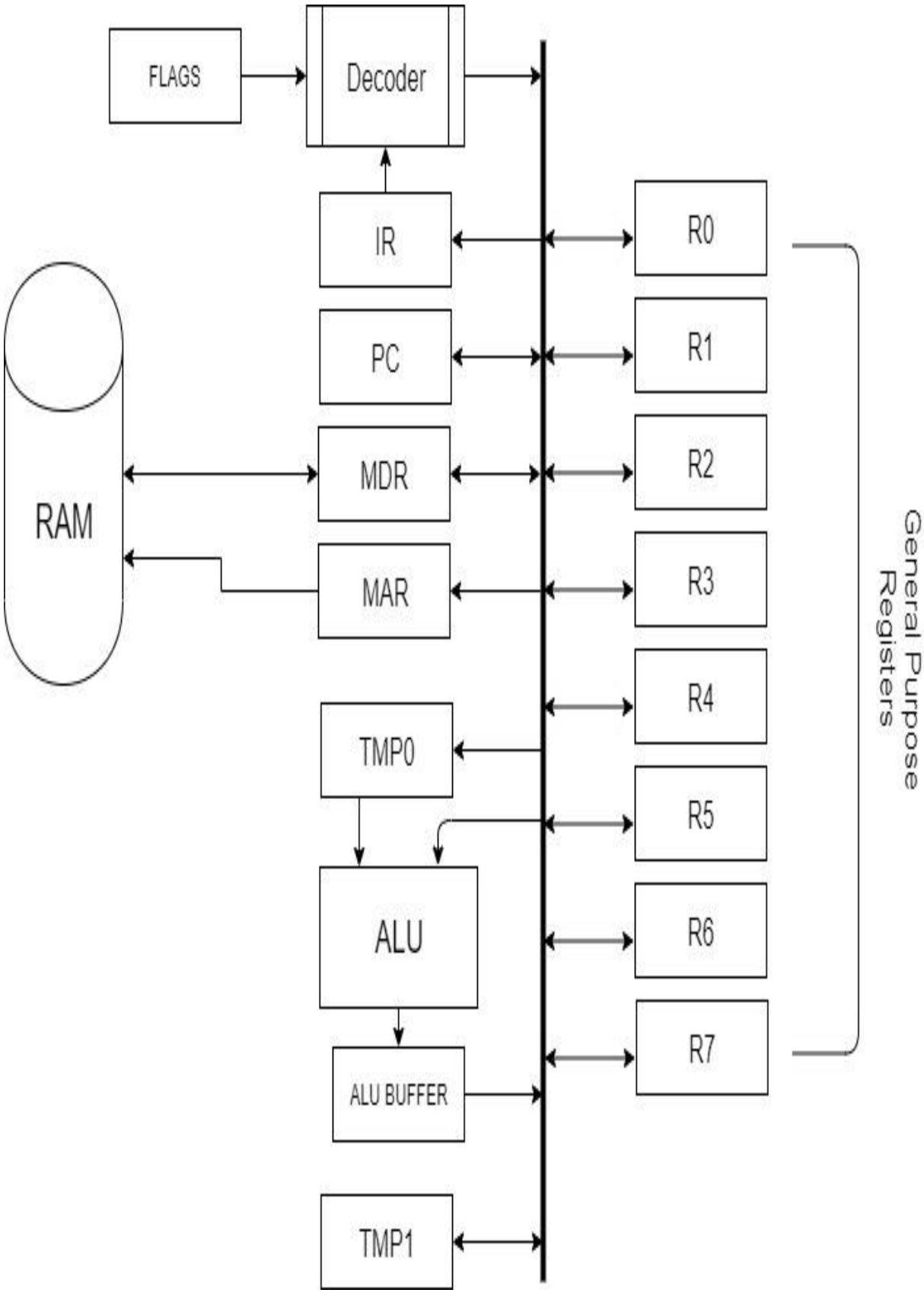
Temp1 : is for **storing** the source value for two operand instructions

Temp0: is for adding **offsets** to the indexed modes

ALU buffering: is for **buffering** the ALU output

#of special registers = **4**(MAR, MDR, IR, FLAG)

#of Busses = 1


FLAGS
Decoder
IR
PC
MDR
MAR
RAM
TMP0
ALU
ALU BUFFER
TMP1
R0
R1
R2
R3
R4
R5
R6
R7
General Purpose Registers