8-Bit CPU Manual

Breadboard Computing

Contents

1 Overview

2 Processor Architecture

2.1 Registers:

- ◇ A Register: `REG_A_0`, `REG_A_1`, . . . , `REG_A_7`
- ◇ B Register: `REG_B_0`, `REG_B_1`, . . . , `REG_B_7`
- ◇ Temporary Register: `REG_TMP_00`, `REG_TMP_01`, . . . , `REG_TMP_15`
- ◇ Flag Register: `REG_F_0`, `REG_F_1`, . . . , `REG_F_7`
- ◇ Instruction Register: `REG_IR_0`, `REG_IR_1`, . . . , `REG_IR_7`

2.2 Buses

- ◇ Data Bus: `BUS_0`, `BUS_1`, . . . , `BUS_7`
- ◇ Address Bus: `ADDRESS_00`, `ADDRESS_01`, . . . , `ADDRESS_15`

2.3 Flags

- ◇ REG_F_0: sign flag(1 → result is negative; 0 → result is non-negative)
- ◇ REG_F_1: parity flag(1 → result is odd; 0 → result is even)
- ◇ REG_F_2: zero flag(1 → result is not equal to zero; 0 → result is equal to zero)
- ◇ REG_F_3: carry flag(for unsigned number operations: 1 → result is out of range; 0 → result is correct)
- ◇ REG_F_4: overflow flag(for signed number operations: 1 → result is out of range; 0 → result is correct)

2.4 Signals

Clock			
CLK		signal generated by clock with specified frequency	
$\overline{\text{CLK}}$		inverted signal generated by clock with specified frequency	

Table 1: Clock Signals Description

Registers			
REG_A_LOAD	A0	load data from data bus to register A	combined with CLK
REG_B_LOAD	A1	load data from data bus to register B	combined with CLK
TMP_H_LOAD	A2	load data from data/address bus to higher bits of temporary register	combined with CLK
TMP_L_LOAD	A3	load data from data/address bus to lower bits of temporary register	combined with CLK
$\overline{\text{TMP\_H\_OUT}}$	A4	pass data from higher bits of temporary register to data/address bus	
$\overline{\text{TMP\_L\_OUT}}$	A5	pass data from lower bits of temporary register to data/address bus	
$\overline{\text{PASS\_DATA}}$	A6	enable passing data from data bus/temporary register to temporary register/data bus	
$\overline{\text{PASS\_ADDRESS}}$	A7	enable passing address from address bus/temporary register to temporary register/address bus	
DATA_DIR	B0	data flow direction selector: pass data from data bus/temporary register to temporary register/data bus	
ADDRESS_DIR	B1	address direction selector: pass data from address bus/temporary register to temporary register/address bus	

Table 2: Registers Signals Description

Arithmetic Logic Unit			
ALU_OPC_0	B2	bit 0 of operation code to perform	
ALU_OPC_1	B3	bit 1 of operation code to perform	
ALU_OPC_2	B4	bit 2 of operation code to perform	
ALU_OPC_3	B5	bit 3 of operation code to perform	
ALU_OPC_4	B6	bit 4 of operation code to perform	
REG_F_LOAD	B7	load flags of current operation to flag register	combined with CLK
$\overline{\text{REG\_F\_OUT}}$	C0	pass data from flag register to data bus	
$\overline{\text{ALU\_OUT}}$	C1	pass current operation result to data bus	

Table 3: Arithmetic Logic Unit Signals Description

Memory Unit			
MAR_LOAD	C2	load data from address bus to memory address register	combined with CLK
MBR_LOAD	C3	load data from data bus to memory buffer register	combined with CLK
$\overline{\text{MAR\_OUT}}$	C4	pass data from memory address register to address bus	
$\overline{\text{MBR\_OUT}}$	C5	pass data from memory buffer register to data bus	
$\overline{\text{MEM\_OUT}}$	C6	read data to memory buffer register from memory at address from memory address register	
$\overline{\text{MEM\_IN}}$	C7	write data from memory buffer register to memory at address from memory address register	
WORD_DIR	D0	data flow direction selector: pass data from data bus/memory buffer register to memory buffer register/data bus	
$\overline{\text{PASS\_WORD}}$	D1	enable passing data from data bus/memory buffer register to memory buffer register/data bus	
MEM_PART	D2	additional bit of memory address; it is used to easy memory division	
$\overline{\text{ZERO\_PAGE}}$	D3	clear higher bits of memory address; it is used to faster access to memory at low addresses	
$\overline{\text{MAN\_INPUT}}$	D4	memory access selector: access to memory address register and memory buffer register by buttons or by buses	

Table 4: Memory Unit Signals Description

2.5 Instructions

2.6 Instruction Cycle

- ◇ Fetch(fixed number of cycles)
- ◇ Decode(variable number of cycles)

Program Counter			
$\overline{\texttt{PC\_LOAD}}$	D5	load address from address bus to program counter	
$\overline{\texttt{PC\_RST}}$		set program counter to 0	
PC_TICK	D6	increment program counter	combined with $\overline{\texttt{CLK}}$
$\overline{\texttt{PC\_OUT}}$	D7	pass data from program counter to address bus	

Table 5: Program Counter Signals Description

Stack Counter			
$\overline{\texttt{STC\_LOAD}}$	E0	load address from address bus to stack counter	
STC_RST		set stack counter to 0	
STC_TICK	E1	increment/decrement stack counter; increment/decrement depends on mode	combined with $\overline{\texttt{CLK}}$
STC_MODE	E2	stack counter operation selector: increment or decrement	
$\overline{\texttt{STC\_OUT}}$	E3	pass data from stack counter to address bus	

Table 6: Stack Counter Signals Description

Control Unit			
IR_LOAD	E4	load instruction from data bus	combined with $\overline{\texttt{CLK}}$
MCC_TICK	E5	increment microcode counter	combined with $\overline{\texttt{CLK}}$
$\overline{\texttt{MCC\_RST}}$	E6	set microcode counter to 0	

Table 7: Control Unit Signals Description

- ◇ Execute(variable number of cycles)

2.7 Decoder

Input: $MI_7I_6I_5I_4I_3I_2I_1I_0C_3C_2C_1C_0$:

- ◇ M - operating mode,
- ◇ $I_7I_6I_5I_4I_3I_2I_1I_0$ - instruction,
- ◇ $C_3C_2C_1C_0$ - microcode counter