

The AVR Microcontroller and Embedded Systems: Using Assembly and C

1st Edition

Contents

CHAPTER 0: INTRODUCTION TO COMPUTING.....	5
SECTION 0.1: NUMBERING AND CODING SYSTEMS	5
SECTION 0.2: DIGITAL PRIMER	6
SECTION 0.3: SEMICONDUCTOR MEMORY	8
SECTION 0.4: CPU AND HARVARD ARCHITECTURE	10
CHAPTER 1: THE AVR MICROCONTROLLERS: HISTORY AND FEATURES.....	11
SECTION 1.1: MICROCONTROLLERS AND EMBEDDED PROCESSORS	11
SECTION 1.2: OVERVIEW OF THE AVR FAMILY	11
CHAPTER 2: AVR ARCHITECTURE & ASSEMBLY LANGUAGE PROGRAMMING.....	13
SECTION 2.1: THE GENERAL PURPOSE REGISTERS IN THE AVR	13
SECTION 2.2: THE AVR DATA MEMORY	13
SECTION 2.3: USING INSTRUCTIONS WITH THE DATA MEMORY	14
SECTION 2.4: AVR STATUS REGISTER	15
SECTION 2.5: AVR DATA FORMAT AND DIRECTIVES	16
SECTION 2.6: INSTRUCTION TO AVR ASSEMBLY PROGRAMMING AND	17
SECTION 2.7: ASSEMBLING AN AVR PROGRAM	17
SECTION 2.8: THE PROGRAM AND PROGRAM ROM SPACE IN THE AVR	18
SECTION 2.9: RISC ARCHITECTURE IN THE AVR	20
CHAPTER 3: BRANCH, CALL AND TIME DELAY LOOP.....	21
SECTION 3.1: BRANCH INSTRUCTIONS AND LOOPING	21
SECTION 3.2: CALL INSTRUCTIONS AND STACK	21
SECTION 3.3: AVR TIME DELAY AND INSTRUCTION PIPELINE	22
CHAPTER 4: AVR I/O PORT PROGRAMMING.....	24
SECTION 4.1: I/O PORT PROGRAMMING IN AVR	24
SECTION 4.2: I/O BIT MANIPULATION PROGRAMMING	25
CHAPTER 5: ARITHMETIC, LOGIC INSTRUCTIONS, AND PROGRAMS.....	29
SECTION 5.1: ARITHMETIC INSTRUCTIONS	29
SECTION 5.2: SIGNED NUMBER CONCEPTS AND ARITHMETIC OPERATIONS	31
SECTION 5.3: LOGIC AND COMPARE INSTRUCTIONS	31
SECTION 5.4: ROTATE AND SHIFT INSTRUCTIONS AND DATA SERIALIZATION	32
SECTION 5.5: BCD AND ASCII CONVERSION	33
CHAPTER 6: AVR ADVANCED ASSEMBLY LANGUAGE PROGRAMMING.....	35
SECTION 6.1: INTRODUCING SOME MORE ASSEMBLER DIRECTIVES	35
SECTION 6.2: REGISTER AND DIRECT ADDRESSING MODES	35
SECTION 6.3: REGISTER INDIRECT ADDRESSING MODE	36
SECTION 6.4: LOOK-UP TABLE AND TABLE PROCESSING	37
SECTION 6.5: BIT-ADDRESSABILITY	40
SECTION 6.6: ACCESSING EEPROM IN AVR	42
SECTION 6.7: CKECKSUM AND ASCII SUBROUTINES	45
SECTION 6.8: MACROS	48
CHAPTER 7: AVR PROGRAMMING IN C.....	49
SECTION 7.1: DATA TYPES AND TIME DELAYS IN C	49
SECTION 7.2: I/O PROGRAMMING IN C	49
SECTION 7.3: LOGIC OPERATIONS IN C	51
SECTION 7.4: DATA CONVERSION PROGRAMS IN C	52
SECTION 7.6: MEMORY ALLOCATION IN C	53
CHAPTER 8: AVR HARDWARE CONNECTION.....	55

SECTION 8.1: ATMEGA32 PIN CONNECTION	55
SECTION 8.2: AVR FUSE BITS	55
SECTION 8.3: EXPLAINING THE INTEL HEX FILE FOR AVR	55
SECTION 8.4: AVR PROGRAMMIN AND TRAINER BOARD	56
CHAPTER 9: AVR TIMER PROGRAMMING IN ASSEMBLY AND C	57
SECTION 9.1: PROGRAMMING TIMERS 0, 1, AND 2	57
SECTION 9.2: COUNTER PROGRAMMING	59
SECTION 9.3: PROGRAMMING TIMERS IN C	60
Chapter 10: INTERRUPT PROGRAMMING IN ASSEMBLY AND C	65
SECTION 10.1: AVR INTERRUPTS	65
SECTION 10.2: PROGRAMMING TIMER INTERRUPTS	65
SECTION 10.3: PROGRAMMING EXTERNAL HARDWARE INTERRUPTS	68
SECTION 10.4: INTERRUPT PRIORITY IN THE AVR	70
CHAPTER 11: AVR SERIAL PORT PROGRAMMING IN ASSEMBLY AND C	71
SECTION 11.1: BASICS OF SERIAL COMMUNICATION	71
SECTION 11.2: ATMEGA32 CONNECTION TO RS232	71
SECTION 11.3: AVR SERIAL PORT PROGRAMMING IN ASSEMBLY	72
SECTION 11.4: AVR SERIAL PORT PROGRAMMING IN C	74
CHAPTER 12: LCD AND KEYBOARD INTERFACING	76
SECTION 12.1: LCD INTERFACING	76
SECTION 12.2: KEYBOARD INTERFACING	76
CHAPTER 13: ADC, DAC, AND SENSOR INTERFACING	77
SECTION 13.1: ADC CHARACTERISTICS	77
SECTION 13.2: ADC PROGRAMMING IN THE ATMEGA32	77
SECTION 13.3: SENSOR INTERFACING AND SIGNAL CONDITIONING	78
SECTION 13.4: DAC INTERFACING	78
Chapter 14: RELAY, OPTOISOLATOR, AND STEPPER MOTOR INTERFACING WITH AVR	80
SECTION 14.1: RELAYS AND OPTOISOLATORS	80
SECTION 14.2: STEPPER MOTOR INTERFACING	80
Chapter 15: INPUT CAPTURE AND WAVE GENERATION IN AVR	81
SECTION 15.1: WAVE GENERATION USING 8-BIT TIMERS	81
SECTION 15.2: WAVE GENERATION USING TIMER1	82
SECTION 15.3: INPUT CAPTURE PROGRAMMING	84
SECTION 15.4: C PROGRAMMING	84
Chapter 16: PWM AND DC MOTOR CONTROL	87
SECTION 16.1: DC MOTOR INTERFACING AND PWM	87
SECTION 16.2: PWM MODES IN 8-BIT TIMERS	87
SECTION 16.3: PWM MODES IN TIMER1	89
Chapter 17: SPI PROTOCOL AND MAX7221 DESPLAY INTERFACING	92
SECTION 17.1: SPI BUS PROTOCOL	92
SECTION 17.2: SPI PROGRAMMING IN AVR	92
SECTION 17.3: MAX7221 INTERFACING AND PROGRAMMING	92
Chapter 18: I2C PROTOCOL AND DS1307 RTC INTERFACING	93
SECTION 18.1: I2C BUS PROTOCOL	93
SECTION 18.2: TWI PROGRAMMING IN AVR	93
SECTION 18.3: AVR TWI PROGRAMMING IN ASSEMBLY AN C	93
SECTION 18.4: DS1307 RTC INTERFACING AND PROGRAMMING	95

CHAPTER 0: INTRODUCTION TO COMPUTING

SECTION 0.1: NUMBERING AND CODING SYSTEMS

1.
 - (a) $12_{10} = 1100_2$
 - (b) $123_{10} = 0111\ 1011_2$
 - (c) $63_{10} = 0011\ 1111_2$
 - (d) $128_{10} = 1000\ 0000_2$
 - (e) $1000_{10} = 0011\ 1110\ 1000_2$
2.
 - (a) $100100_2 = 36_{10}$
 - (b) $1000001_2 = 65_{10}$
 - (c) $11101_2 = 29_{10}$
 - (d) $1010_2 = 10_{10}$
 - (e) $00100010_2 = 34_{10}$
3.
 - (a) $100100_2 = 24_{16}$
 - (b) $1000001_2 = 41_{16}$
 - (c) $11101_2 = 1D_{16}$
 - (d) $1010_2 = 0A_{16}$
 - (e) $00100010_2 = 22_{16}$
4.
 - (a) $2B9_{16} = 0010\ 1011\ 1001_2, 697_{10}$
 - (b) $F44_{16} = 1111\ 0100\ 0100_2, 3908_{10}$
 - (c) $912_{16} = 1001\ 0001\ 0010_2, 2322_{10}$
 - (d) $2B_{16} = 0010\ 1011_2, 43_{10}$
 - (e) $FFFF_{16} = 1111\ 1111\ 1111\ 1111_2, 65535_{10}$
5.
 - (a) $12_{10} = 0C_{16}$
 - (b) $123_{10} = 7B_{16}$
 - (c) $63_{10} = 3F_{16}$
 - (d) $128_{10} = 80_{16}$
 - (e) $1000_{10} = 3E8_{16}$
6.
 - (a) $1001010 \rightarrow 0011\ 0110$
 - (b) $111001 \rightarrow 0000\ 0111$
 - (c) $10000010 \rightarrow 0111\ 1110$
 - (d) $111110001 \rightarrow 0000\ 1111$
7.
 - (a) $2C+3F = 6B$
 - (b) $F34+5D6 = 150A$
 - (c) $20000+12FF = 212FF$
 - (d) $FFFF+2222 = 12221$
8.
 - (a) $24F-129 = 126_{16}$

- (b) $FE9-5CC = A1D_{16}$
- (c) $2FFFF-FFFF = 30000_{16}$
- (d) $9FF25-4DD99 = 5218C_{16}$

9.

- (a) Hex: 30, 31, 32, 33, 34, 35, 36, 37, 38, 39
- (b) Binary: 11 0000, 11 0001, 11 0010, 11 0011, 11 0100, 11 0101, 11 0110, 11 0111, 11 1000, 11 1001.

	<u>ASCII (hex)</u>	<u>Binary</u>	
0	30	011	0000
1	31	011	0001
2	32	011	0010
3	33	011	0011
4	34	011	0100
5	35	011	0101
6	36	011	0110
7	37	011	0111
8	38	011	1000
9	39	011	1001

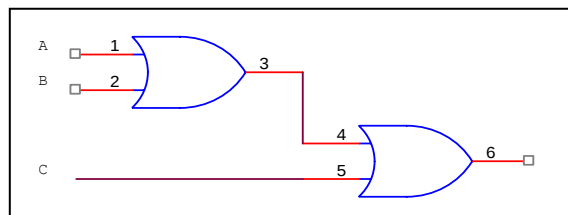
10.

000000 22 55 2E 53 2E 41 2E 20 69 73 20 61 20 63 6F 75
000010 6E 74 72 79 22 0D 0A 22 69 6E 20 4E 6F 72 74 68
000020 20 41 6D 65 72 69 63 61 22 0D 0A

"U.S.A. is a cou
ntry"..in North
America"..

SECTION 0.2: DIGITAL PRIMER

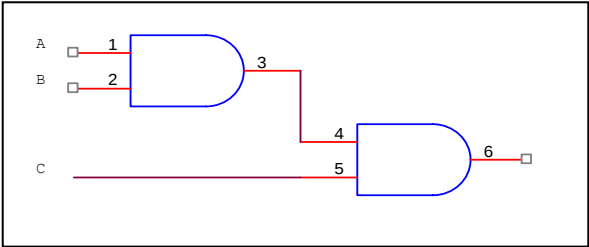
11.



12.

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

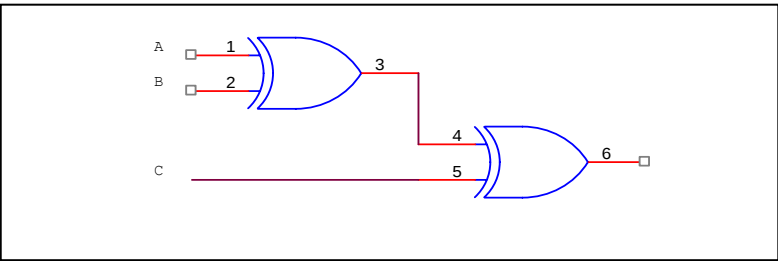
13.



14.

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

15.



A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

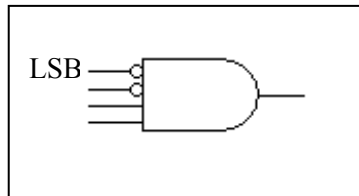
16.

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

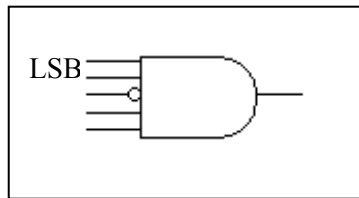
17.

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

18.



19.



20.

CLK	D	Q
No	X	NC
Yes	0	0
Yes	1	1

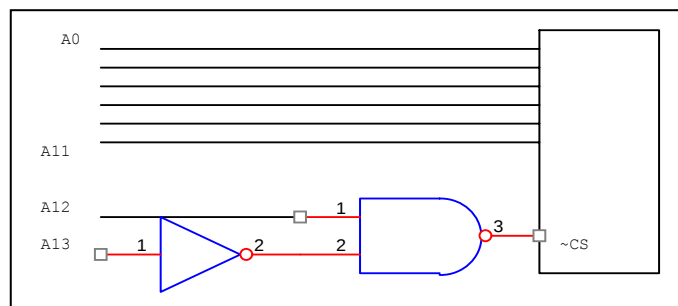
SECTION 0.3: SEMICONDUCTOR MEMORY

21.

- (a) 4
- (b) 4
- (c) 4
- (d) 1 048 576, 2^{20}
- (e) 1024K
- (f) 1 073 741 824, 2^{30}
- (g) 1 048 576 K
- (h) 1024M
- (i) 8388608, 8192K

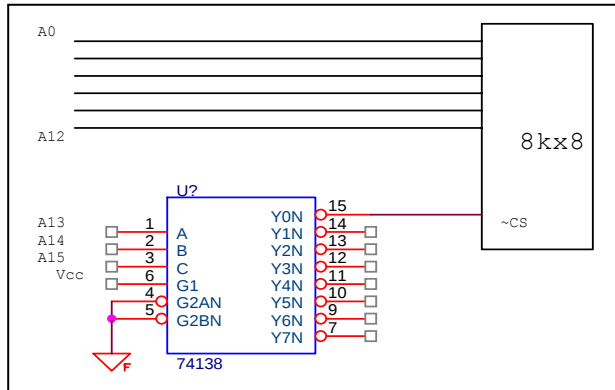
22. Disk storage capacity / size of a page = $(2 \cdot 2^{30}) / (25 \cdot 80) = 1$ million pages

23. (a) $9\text{FFFFh} - 10000\text{h} = 8\text{FFFFh} = 589\,824$ bytes
 (b) 576 kbytes
24. $2^{32} - 1 = 4\,294\,967\,295$
25. (a) FFh, 255
 (b) FFFFh, 65535
 (c) FFFF FFFFh, 4 294 967 295
 (d) FFFF FFFF FFFF FFFFh, 18 446 744 073 709 551 615
26. (a) $2^{16} = 64\text{K}$
 (b) $2^{24} = 16\text{M}$
 (c) $2^{32} = 4096$ Mega, 4G
 (d) $2^{48} = 256$ Tera, 262144 Giga, 268435456 Mega
27. Data bus is bidirectional, address bus is unidirectional (exit CPU).
28. The storage of the chip is measured in Megabits while the Computer memory is measured in Megabytes.
29. True, the more address lines the more memory locations.
30. True, the memory location size is fixed.
31. True, the more data lines the more memory locations
32. True
33. access time
34. True
35. electrically erasable
36. True
37. DRAM
38. SRAM
39. DRAM and SRAM
40. (c)
41. (c)
42. (a) 32Kx8, 256K (f) 8Kx1, 8K
 (b) 8Kx8, 64K (g) 4Kx8, 32K
 (c) 4Kx8, 32K (h) 2Kx8, 16K
 (d) 8Kx8, 64K (i) 256Kx4, 1M
 (e) 4Mx1, 4M (j) 64Kx8, 512K
43. (a) 128K 14 8 (f) 256K 8 4
 (b) 256K 15 8 (g) 8M 20 8
 (c) 512K 16 8 (h) 16M 11 4
 (d) 2M 18 8 (i) 512K 16 8
 (e) 512K 16 8
44. 4000h - 7FFFh
- 45.



46. 8000h - 8FFFh, B000h - BFFFh, E000 - EFFFh

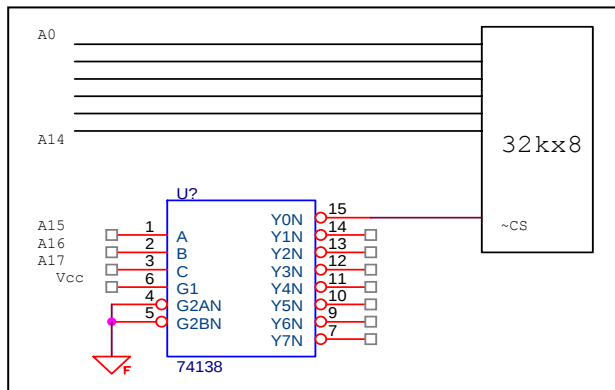
47.



Each controls 8K bytes block

48. 6000h - 7FFFh, C000h - DFFFh, E000h - FFFFh

49.



Each controls 32K bytes block

50. 4000h - 7FFFh, 8000h - BFFFh, C000h - FFFFh

SECTION 0.4: CPU AND HARVARD ARCHITECTURE

51. PC (Program Counter)

52. ALU is responsible for all arithmetic and logic calculations in the CPU.

53. Address, control and data

CHAPTER 1: THE AVR MICROCONTROLLERS: HISTORY AND FEATURES

SECTION 1.1: MICROCONTROLLERS AND EMBEDDED PROCESSORS

1. False.
2. True.
3. True.
4. True.
5. CPU, RAM, ROM, EEPROM, I/O, Timer, Serial COM port, ADC.
6. RAM and ROM.
7. Keyboard, mouse, printer.
8. Computing power and compatibility with millions and millions of PCs.
9. PIC 16x – Microchip Technology, 8051 - Intel, AVR – Atmel, Z8 – Zilog, 68HC11 – Freescale Semiconductor (Motorola).
10. 8051.
11. Power consumption.
12. The ROM area is where the executable code is stored.
13. Very, in case there is a shortage by one supplier.
14. Suppliers other than the manufacturer of the chip.
15. B is absolutely wrong; 16 bit software can not run on an 8 bit system due to special instructions and registers. But A can be true (in the case of software compatibility).

SECTION 1.2: OVERVIEW OF THE AVR FAMILY

16. Flash memory is ideal for fast development because Flash memory can be erased in seconds compared to 20 minutes or more needed for the UV-EPROM.
17. 32 pins
18. 32 Kbytes
19. 256 bytes
20. 8
21. 4K
22. 6
23. 2K
24. ATtiny11, ATtiny12, ATtiny15L
25. All AVRs have Flash ROM; but a few of them, like ATtiny20, ATtiny28L, and ATtiny4 have no EEPROM
26. AT90USB1286, AT90USB1287, AT90USB162, AT90USB646, AT90USB647, AT90USB82
27. AT90CAN128, AT90CAN32, AT90CAN64
- 28.

	Program ROM	Data RAM
(a) ATmega32	32 Kbytes	2048
(b) ATtiny44	2 Kbytes	128

(c) ATtiny84	8 Kbytes	512
(d) 90CAN128	128 Kbytes	4 K

29.

	Program ROM	Data RAM	EEPROM
ATmega32	32 Kbytes	2048	1024
ATmega16	16 Kbytes	1024	512

30. 512