ECE 375 PRELAB 8

Lab Time: Friday 16:00 ~ 17:50

Hyunjae Kim

QUESTIONS

1. In this lab, you will be given a set of behaviors/actions that you need to have a proof-of-concept "toy" perform. Think of a toy you know of (or look around online for a toy) that is likely implemented using a microcontroller, and describe the behaviors it performs. Here is an example behavior: "If you press button X on the toy, it takes action Y (or makes sound Z)".

Batman Car Toy

(Image From: https://www.geekalerts.com/batman-dark-knight-tumbler-batmobile-rtr-electric-rc-car/)

If you press the front side of vertical button of remote control, the car goes forward, and if you press the back side of vertical button, the car goes backward.

If you press the right side of horizontal button of remote control, the front wheel turns right, and if you press the left side of horizontal button of remote control, the front wheel turns left.

2. For each behavior you described in the previous question, explain which microcontroller feature was likely used to implement that behavior, and give a brief code example indicating how that feature should be configured. Make your explanation as ATmega128-specific as possible (e.g., discuss which I/O registers would need to be configured, and if any interrupts will be used), and also mention if any additional mechanical and/or electronic devices are needed.

(a) The USART feature of ATmega 128 was likely used to implement the move of the Batman car. The reason is that USART feature enables the board to communicate with external devices connected to the board in wire or wireless.

(b)

Assumption:

There is a remote communication module is connected to USART0 of ATmega128 board. The board only needs to receive data, and do the action. Normal asynchronous mode with 9600 Baud rate.

(Code Example Below)

```
.include        “m128def.inc”
.def            MPR = r16
.org            $0000
                RJMP            INITIALIZE
.ORG            $0024
                RJMP            RECEIVEDATA
.ORG            $0046
INITIALIZE:
                LDI             MPR, LOW(RAMEND)
                OUT             SPL, MPR
                LDI             MPR, HIGH(RAMEND)
                OUT             SPH, MPR
;INITIALIZE USART0 RECEIVE
                LDI             MPR, $00
                OUT             DDRE, MPR
;INITIALIZE MOTOR
                LDI             MPR, $0F
                OUT             DDRB, MPR
                LDI             MPR, 103
                STS             UBBR0, MPR
;Asynchronous, 8-bit data frame, 2 stop bit
                LDI             MPR, (0 << UMSEL0 | 1 << USBS0 | 1 << USCZ01 | 1 << USCZ00)
;Enable receive, set receive interrupt.
                LDI             MPR, (1 << RXEN0 | 1 << RXCIE0)
                OUT             UCSR0B, MPR
                SEI
RECEIVEDATA:
                IN              R17, UDR0
                ;DO MOTOR CONTROL
                RETI
```

3. Each ATmega128 USART module has two flags used to indicate its current transmitter state: the Data Register Empty (UDRE) flag and Transmit Complete (TXC) flag. What is the difference between these two flags, and which one always gets set first as the transmitter runs? You will probably need to read about the Data Transmission process in the datasheet (including looking at any relevant USART diagrams) to answer this question.

(a) The difference

Data Register Empty (UDRE) flag:

UDRE flag indicates the transmit buffer is ready to receive new data. When transmit buffer is empty, the flag is set, and when transmit buffer contains data, the flag is cleared.

Transmit Complete (TXC) flag:

TXC flag is set when there is no data in the transmit buffer and the entire frame in Transmit Shift Register has been shifted out. The flag is automatically cleared when transmit complete interrupt occurs or by writing 0 directly to the flag.

(b) Which flag gets set first as the transmitter runs?

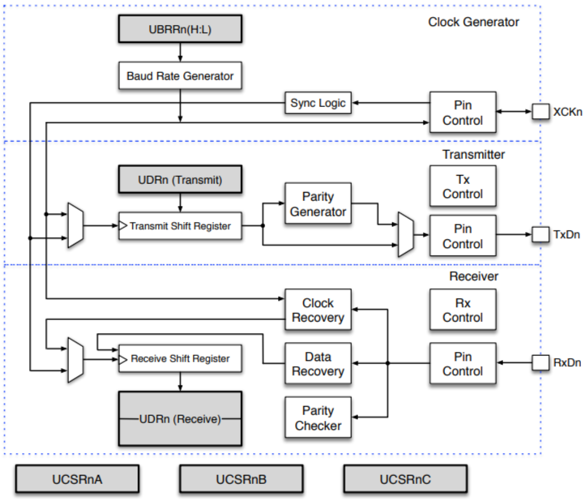
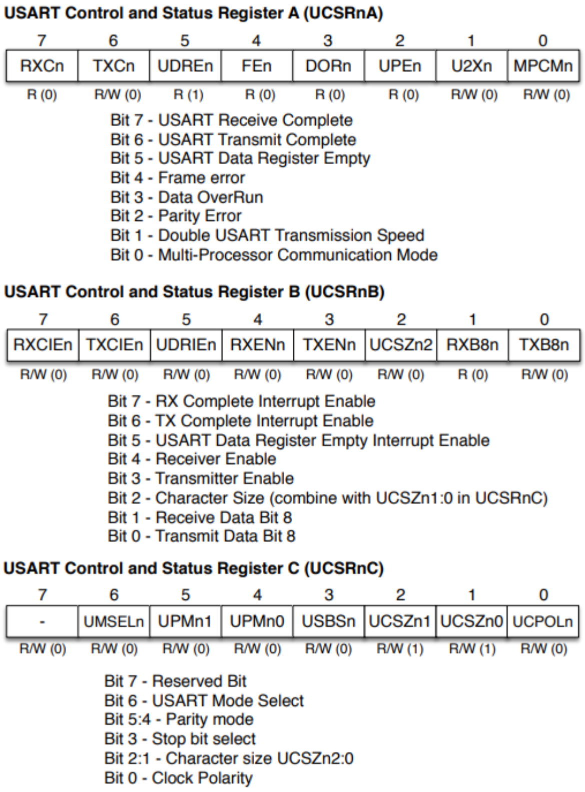


Figure 5.28: USART Block Diagram.

UDRE flag gets set first. Whenever the UDR register is empty, UDRE flag is set, but in case of TXC flag, the Transmit Shift Register has to be empty in order to set the flag. Thus, because of the structure of USART, UDRE flag gets set first.

4. Each ATmega128 USART module has one flag used to indicate its current receiver state (not including the error flags). For USART1 specifically, what is the name of this flag, and what is the interrupt vector address for the interrupt associated with this flag? This time, you will probably need to read about Data Reception in the datasheet to answer this question.



USART Control and Status Register A (UCSRnA)

7	6	5	4	3	2	1	0
RXCn	TXCn	UDREn	FEn	DORn	UPEn	U2Xn	MPCMn
R (0)	R/W (0)	R (1)	R (0)	R (0)	R (0)	R/W (0)	R/W (0)

Bit 7 - USART Receive Complete
Bit 6 - USART Transmit Complete
Bit 5 - USART Data Register Empty
Bit 4 - Frame error
Bit 3 - Data OverRun
Bit 2 - Parity Error
Bit 1 - Double USART Transmission Speed
Bit 0 - Multi-Processor Communication Mode

USART Control and Status Register B (UCSRnB)

7	6	5	4	3	2	1	0
RXCIEn	TXCIEn	UDRIEn	RXENn	TXENn	UCSZn2	RXB8n	TXB8n
R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R (0)	R/W (0)

Bit 7 - RX Complete Interrupt Enable
Bit 6 - TX Complete Interrupt Enable
Bit 5 - USART Data Register Empty Interrupt Enable
Bit 4 - Receiver Enable
Bit 3 - Transmitter Enable
Bit 2 - Character Size (combine with UCSZn1:0 in UCSRnC)
Bit 1 - Receive Data Bit 8
Bit 0 - Transmit Data Bit 8

USART Control and Status Register C (UCSRnC)

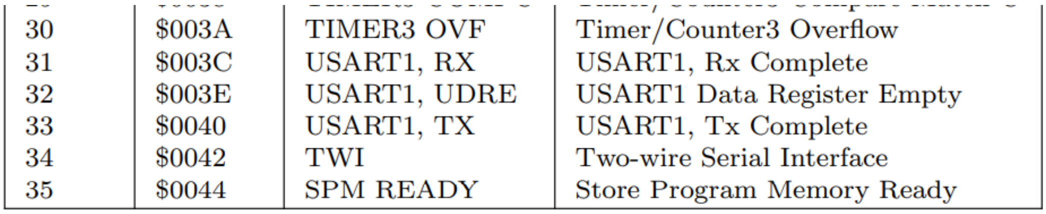
7	6	5	4	3	2	1	0
-	UMSELn	UPMn1	UPMn0	USBSn	UCSZn1	UCSZn0	UCPOLn
R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (1)	R/W (1)	R/W (0)

Bit 7 - Reserved Bit
Bit 6 - USART Mode Select
Bit 5:4 - Parity mode
Bit 3 - Stop bit select
Bit 2:1 - Character size UCSZn2:0
Bit 0 - Clock Polarity

Figure 5.29: USARTn Control and Status Register A, B, and C.

(a) The name of flag is Receive Complete (RXC1) flag in USART 1.

(b) The address of interrupt vector related to this flag is $3C.



30	$003A	TIMER3 OVF	Timer/Counter3 Overflow
31	$003C	USART1, RX	USART1, Rx Complete
32	$003E	USART1, UDRE	USART1 Data Register Empty
33	$0040	USART1, TX	USART1, Tx Complete
34	$0042	TWI	Two-wire Serial Interface
35	$0044	SPM READY	Store Program Memory Ready

(Computer Organization and Assembly Programming: Embedded System Perspectives, Table5.1)

REFERENCE

AVR Datasheet

Computer Organization and Assembly Programming: Embedded System Perspectives, Ben Lee