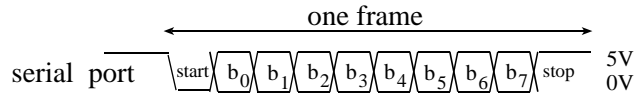


Recap

- Serial communication; what does the frame look like
- SCI shift register versus SCI data register
- How RDRF is set; how RDRF is cleared
- How TDRE is set; how TDRE is cleared

Overview

- Baud rate vs bandwidth
- Latency vs real time
- SCI interrupts
- Introduction to Fifo queue



Performance issues

- Bandwidth (information/sec)
- Latency (time from RDRF to read SCI1DRL)
(time from TDRE to write SCI1DRL)
- Noise immunity (error detection, error correction)
Probability data will be transmitted without error

The **baud rate** is the total number of bits (information, overhead, and idle) per time that are transmitted in the serial communication.

$$\text{baud rate} = \frac{1}{\text{bit time}}$$

The **bandwidth, bit rate** and **throughput** is the number of information bits per time that are transmitted.

$$\text{Bandwidth} = \frac{\text{number of information bits/frame}}{\text{total number of bits/frame}} \cdot \text{baud rate}$$

Receiving serial data requires a real-time interface

If there is already data in the SCI0DRL when the shift register is finished, it will wait until the previous frame is read by the software, before it is transferred. An overrun occurs when there is one receive frame in the SCI0DRL, one receive frame in the receive shift register, and a third frame comes into RxD.

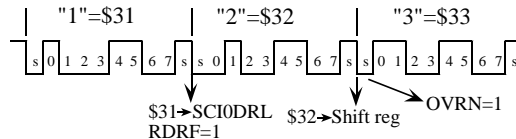


Figure 8.5. Three receive data frames result in an overrun (OR) error.

8.2.4. 6812 SCI Details

address	msb														lsb	Name	
\$00D0	-	-	-	12	11	10	9	8	7	6	5	4	3	2	1	0	SCI1BD
Address	Bit 7		6	5		4	3		2	1		Bit 0		Name			
\$00D2	LOOPS	SWAI	RSRC	M	WAKE	ILT	PE	PT	SCI1CR1								
\$00D3	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCI1CR2								
\$00D4	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF	SCI1SR1								
\$00D5	0	0	0	0	0	BRK13	TXDIR	RAF	SCI1SR2								
\$00D6	R8	T8	0	0	0	0	0	0	SCI1DRH								
\$00D7	R7T7	R6T6	R5T5	R4T4	R3T3	R2T2	R1T1	R0T0	SCI1DRL								

Similar to Table 8.4. 9S12 SCI ports (SCI1).

Run the Tut2 system

- Busy-wait SCI input
- I/O bound (bandwidth limited to input rate)
- Very inefficient (spends a lot of time waiting)

Two ways to recover wasted time

- 1) Combine all busy-waits

Requires cooperation,
Complicated to test (all tasks are interrelated)
Can't guarantee bound on latency

- 2) Input device interrupts when ready
ISR reads new input, puts into FIFO
Latency equals maximum time it runs with I=1

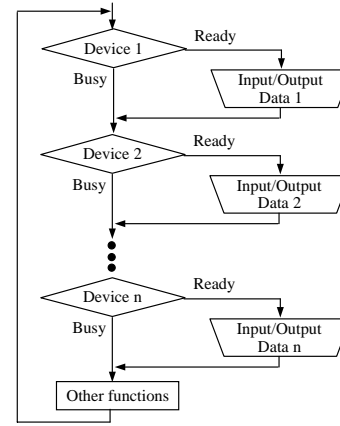
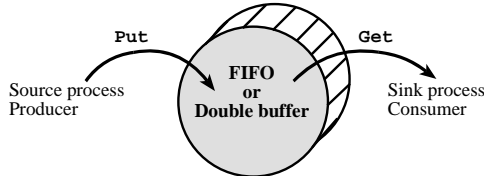


Figure 12.4. FIFO queues and double buffers can be used to pass data from a producer to a consumer.

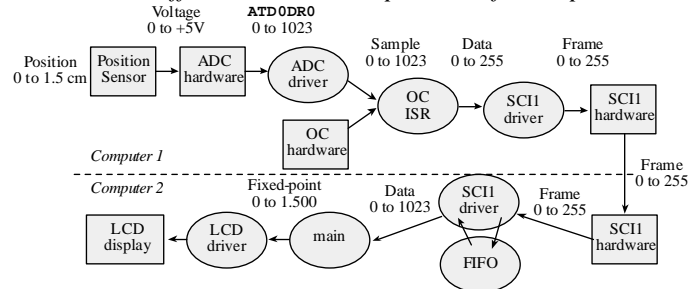


Figure 7.1. Data flows from the sensor through the two microcontrollers to the LCD. The output compare timer is used to trigger the real-time sampling. Use the special serial cable to connect the two SCI1 ports.

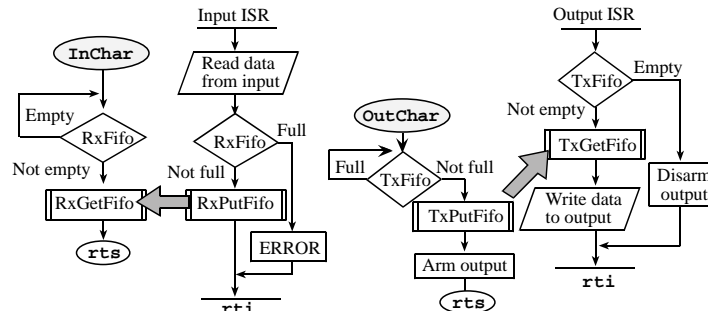


Figure 12.6. FIFO queues can be used to pass data between threads.

Show code for tut4 project

Look at Lab 7, draw flowcharts for the two systems
You can debug each microcontroller separately in TExaS
Show running Lab 7, simulated in TExaS

The bottom line

**Baud rate, bandwidth, latency, real time
SCI interrupts**