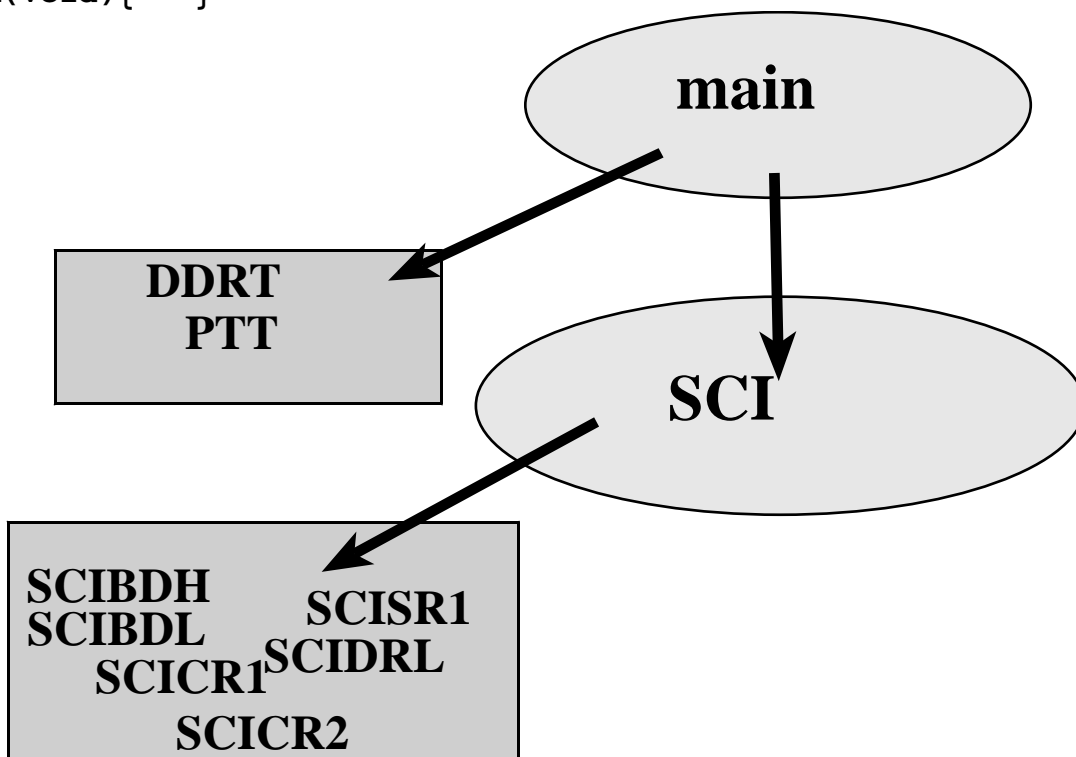


Objectives

- **Parallel processing, distributed processing, multithreading**
- **Modular programming**
- **Call graphs,**
- **Flow charts,**
- **Data flow graphs,**
- **Device drivers: SCI2.C and SCI2A.C,**
- **Metrowerks compiler,**
- **Quality software**

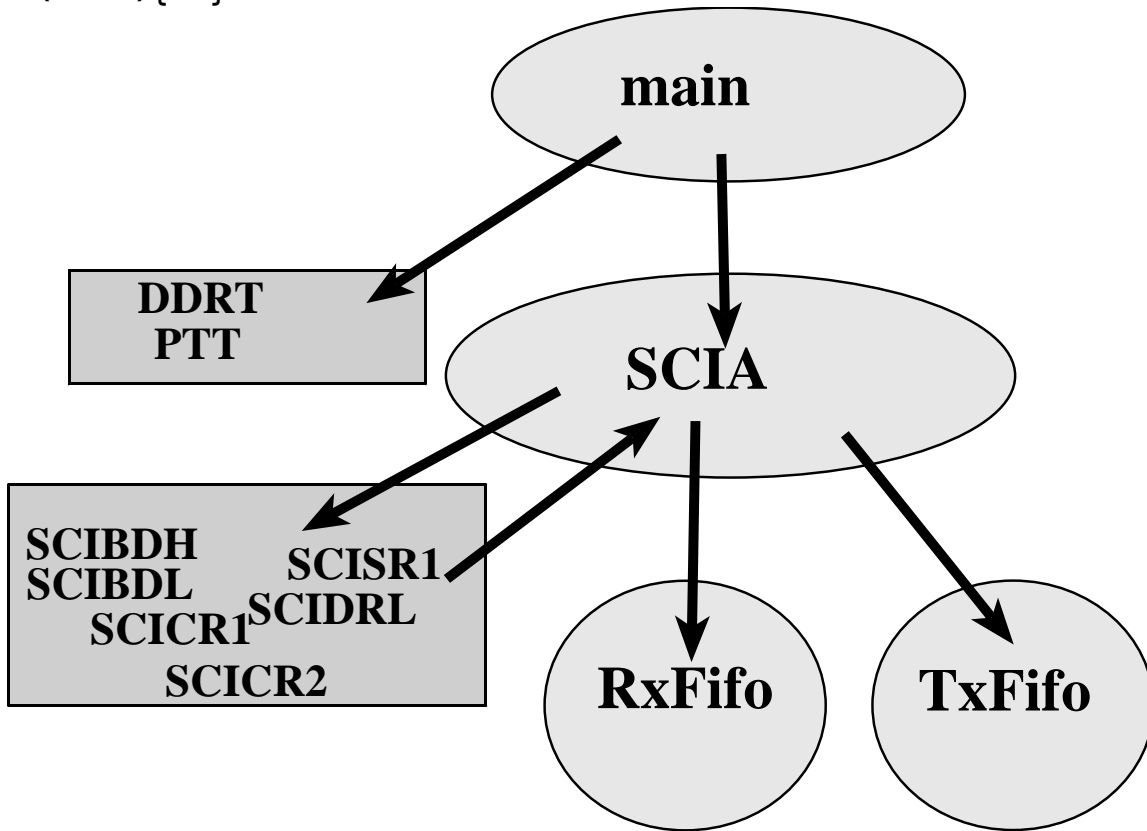
```
// Open Metrowerks SCI project
#include "SCI.H"
void main(void){ }
```



```
// ***** SCI.C *****
#define RDRF 0x20 // private constant
// implementations of public functions
char SCI_InChar(void){
  while((SCISR1 & RDRF) == 0){};
  return(SCIDRL);
}
//-----SCI_OutChar-----
void SCI_OutChar(char data){
  while((SCISR1 & TDRE) == 0){};
  SCIDRL = data; }

// Open Metrowerks Project SCIA
```

```
#include "SCIA.H"
void main(void){ }
```



```
// filename ***** SCIA.C *****
#include "RxFifo.h"
#include "TxFifo.h"
#define TDRE 0x80 // private constant
void SCI_Init(unsigned short br){} // public
interrupt 20 void SciHandler(void){ } // private
```

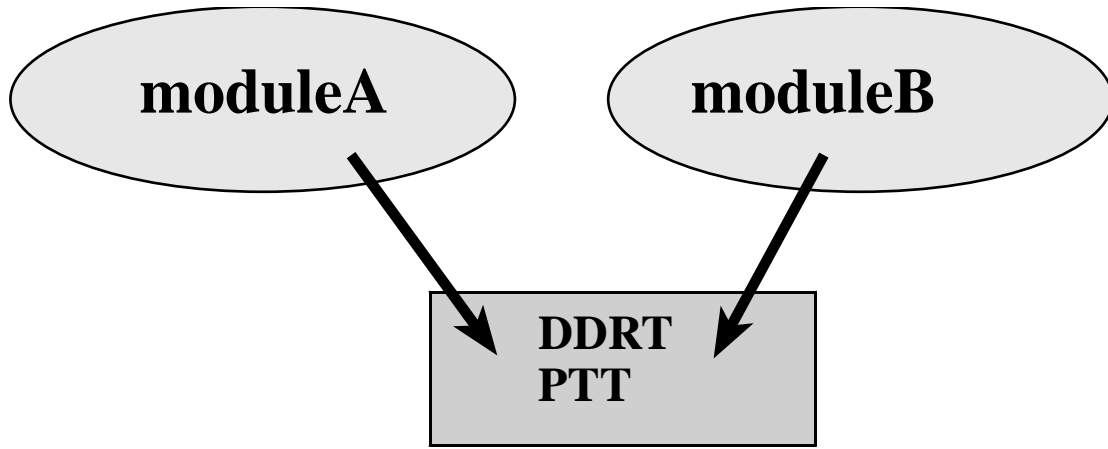
Why private versus public?

- Information hiding
- Reduce coupling
- Separate mechanisms from policy
- Essence of modular design

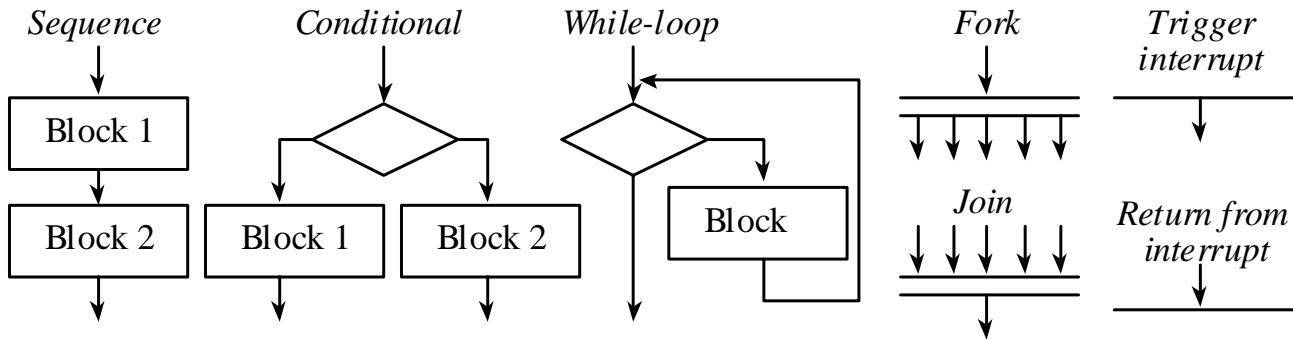
How in C

- Public name has *Module Name* and underline
- Public object has Prototype in header file
- Private globals have **static** modifier

Use call graphs to identify potential conflicts

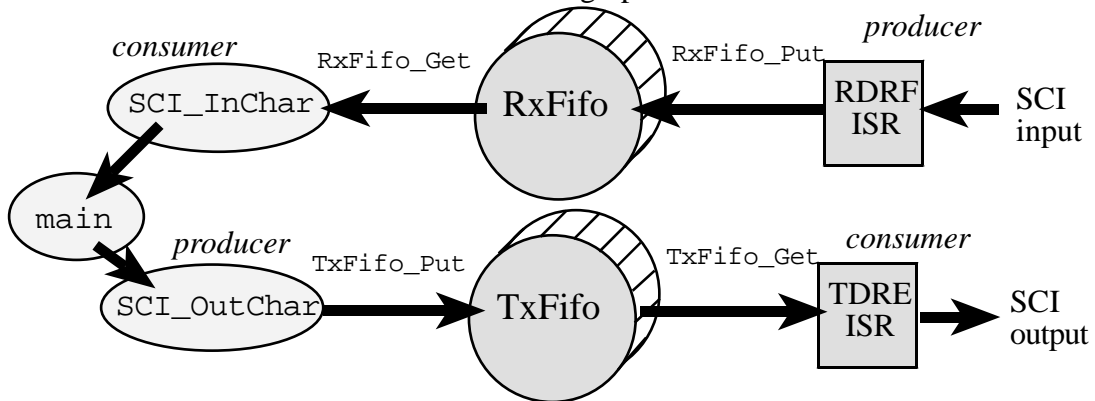


Flowcharts



Flowchart showing the basic building blocks of structured programming.

Data Flow graphs



A data flow graph showing two FIFOs that buffer data between producers and consumers.

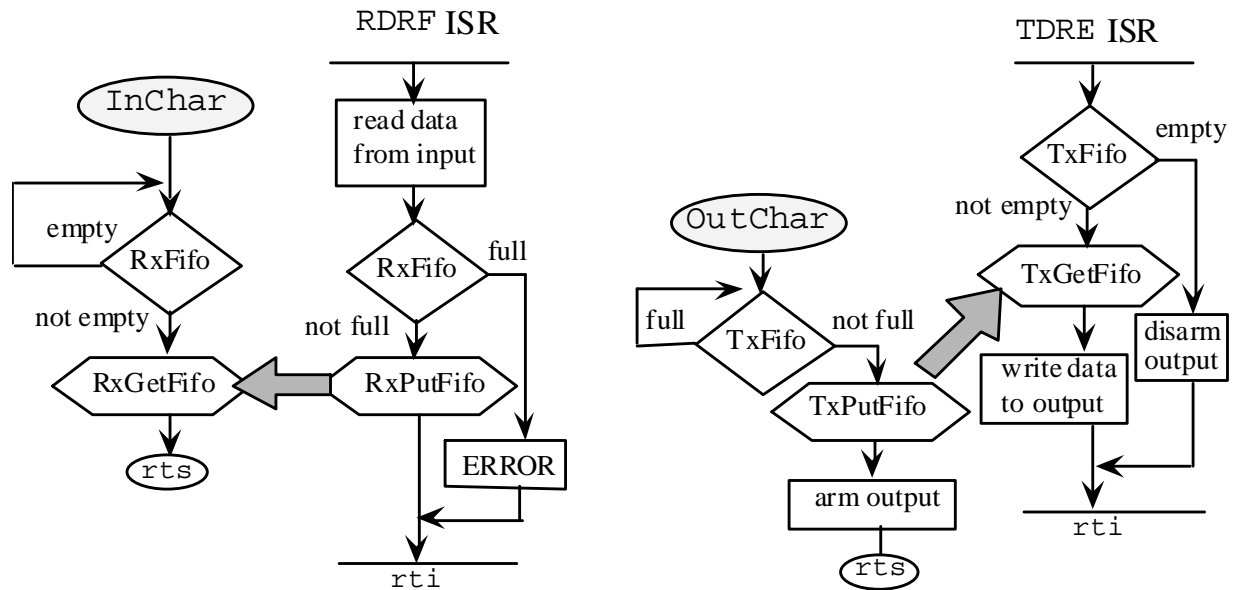


Figure 4.4. FIFO queues can be used to pass data between threads.

```
//-----SCI_InChar-----
// Wait for new serial port input, interrupts
// Input: none
// Output: ASCII code for key typed
char SCI_InChar(void){
char letter;
  while (Rx Fifo_Get(&letter) == 0){};
  return(letter);
}

//-----SCI_OutChar-----
// Wait for buffer to be empty, output to SCI
// interrupt synchronization
// Input: 8-bit data to be transferred
// Output: none
void SCI_OutChar(char data){
  while (Tx Fifo_Put(data) == 0){};
  // spin if Tx Fifo is full
  SCICR2 = 0xAC; /* arm TDRE */
}
/*-----SciHandler-----*/
// RDRF set on new receive data
// TDRE set on an empty transmit data register
interrupt 20 void SciHandler(void){ char data;
  if(SCISR1 & RDRF){
    Rx Fifo_Put(SCIDRL); // clears RDRF
  }
  if((SCICR2&0x80)&&(SCISR1 & TDRE)){
```

```

if(TxFifo_Get(&data)){
    SCIDRL = data; // clears TDRE
}
else{
    SCICR2 = 0x2c; // disarm TDRE
}
}
}
    
```

I/O bound input device

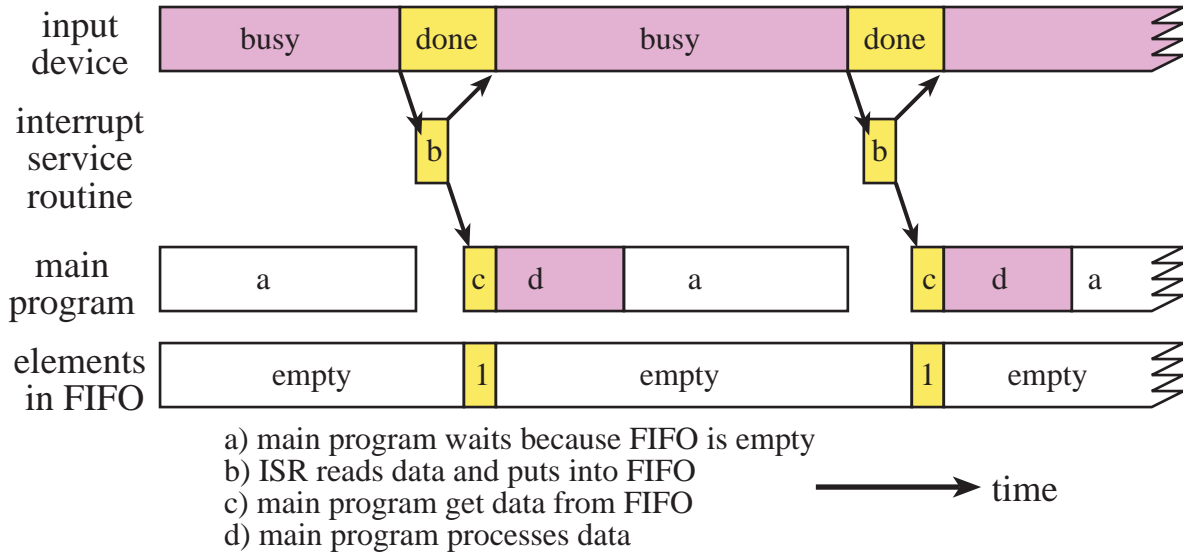


Figure 4.6. Hardware/software timing of an I/O bound input interface.

I/O bound output device (buffered I/O)

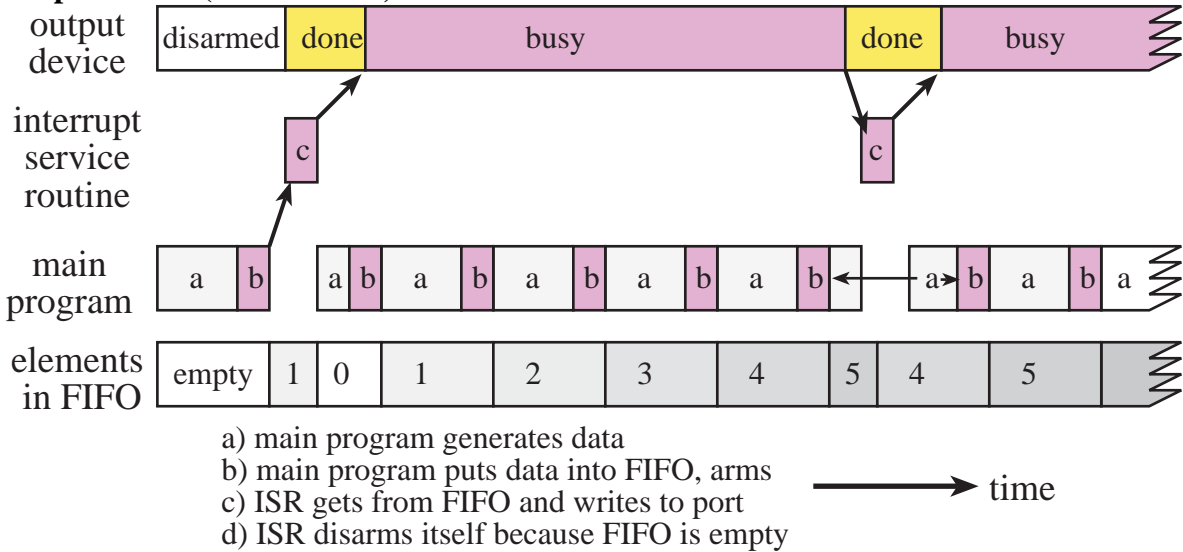
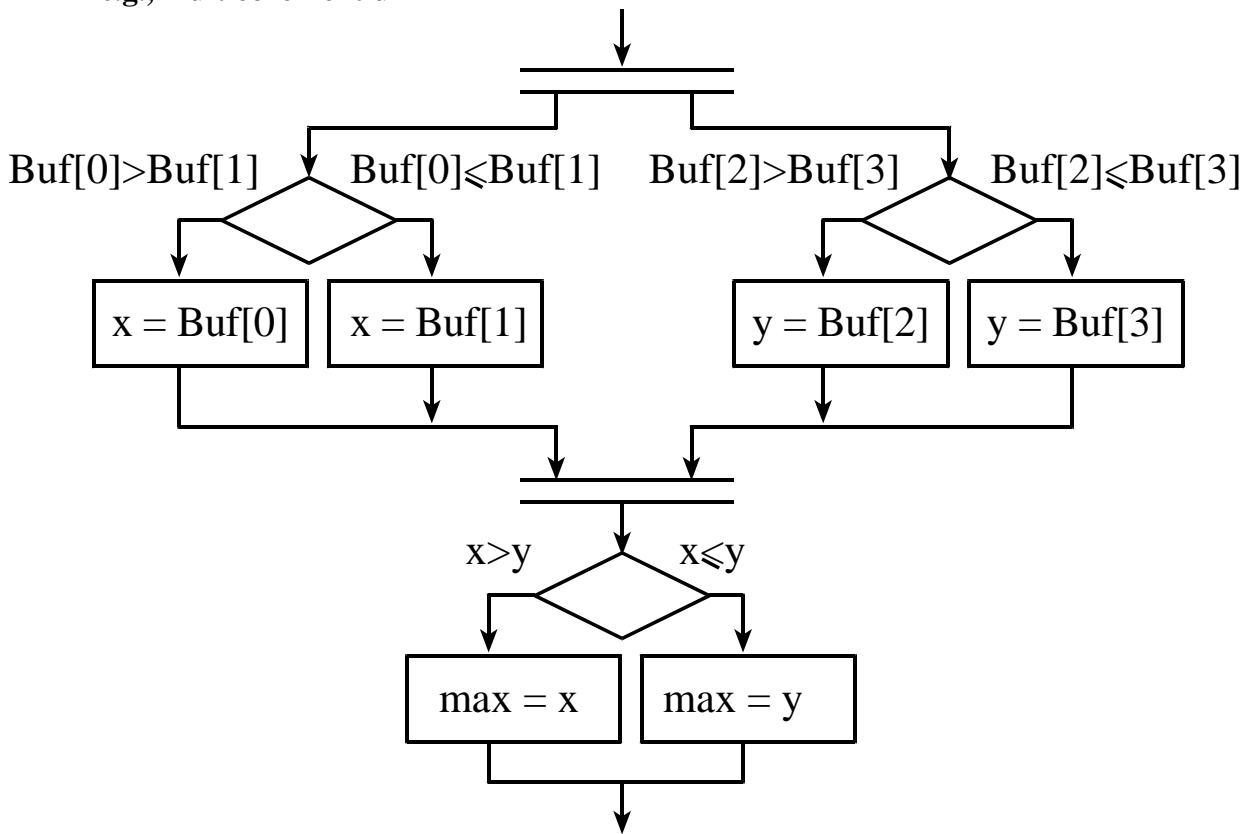


Figure 4.10. Hardware/software timing of an I/O bound output interface.

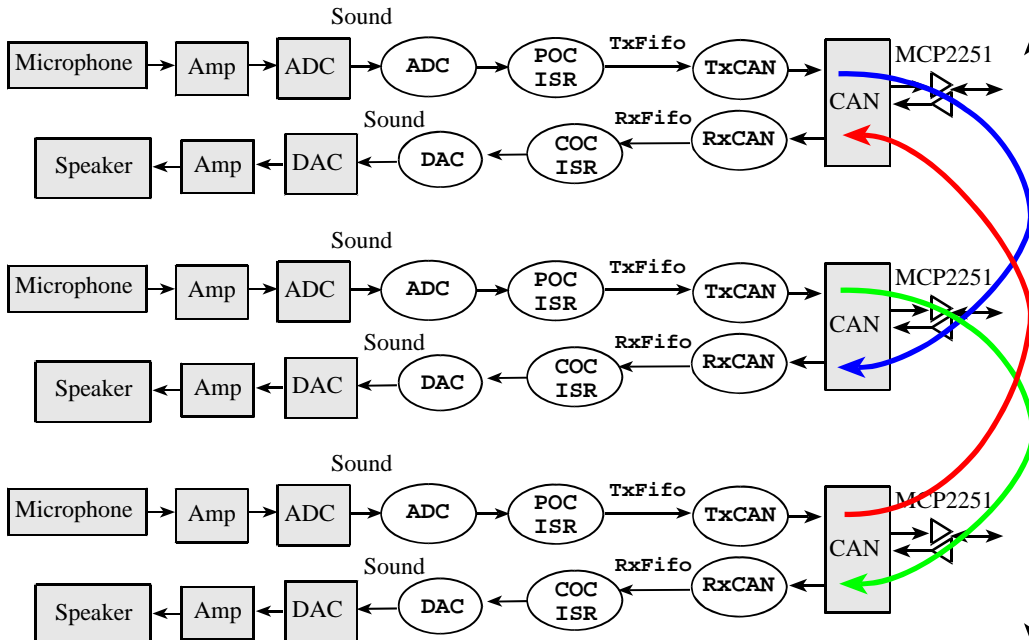
Parallel processing:

multiple processors, shared memory
 simultaneous execution of two or more software tasks
 e.g., multicore Pentium



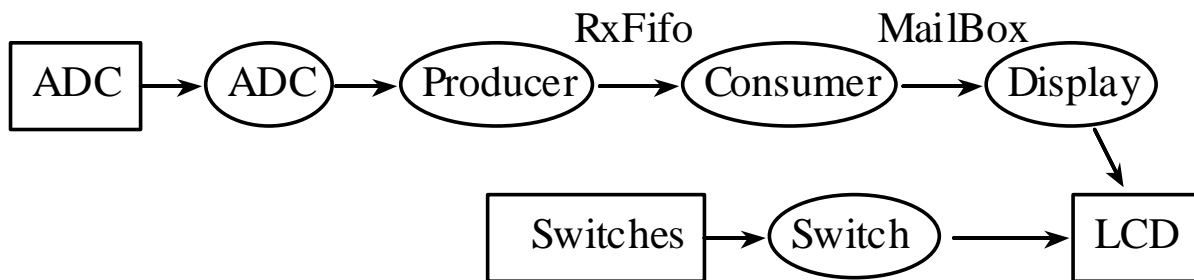
Distributed processing:

multiple computers, separate memory, I/O network link
 simultaneous execution of two or more software tasks
 e.g., Lab 3g



Multithreading

- One foreground and multiple background threads
- Multiple foreground threads using a thread scheduler



Run Interrupting SCI project

- 1) Install Metrowerks
- 2) Get the educational license
- 3) Download SCI project from <http://users.ece.utexas.edu/~valvano/Starterfiles/>
- 4) Connect to PC, Boot mode, reset
- 5) Unzip, open project in Metrowerks, Project->Debug

Request samples (DIP package)

See Course Description page for latest information
 Analog Devices <http://www.analog.com/en/index.html>

- 1) AD8032ANZ rail-to-rail op amp

Maxim <http://www.maxim-ic.com/> (with or without +)

- 1) MAX1247ACPE+ 12-bit ADC (A or B, with or without +)
- 2) MAX6225ACPA+ 2.500V analog reference (ACPA or BCPA)

3a) MAX5154ACPE dual 12-bit SPI interface DAC (ACPE or BCPE)
or 3b) MAX539ACPA single 12-bit SPI interface DAC (ACPA or BCPA)

[Texas Instruments](http://www.ti.com) <http://www.ti.com> (with or without A)

- 1) INA122P rail-to-rail instrumentation amp
- 2) OPA2350PA rail-to-rail dual op amp
- 3) TLC2272ACP rail-to-rail dual op amp
- 4) TLC2274ACN rail-to-rail quad op amp