

ELEC2041

Microprocessors and Interfacing

Lectures 29: I/O Interfacing Examples

<http://webct.edtec.unsw.edu.au/>

May 2006

Saeid Nooshabadi

saeid@unsw.edu.au

ELEC2041 lec29-io-examples.1

Saeid Nooshabadi

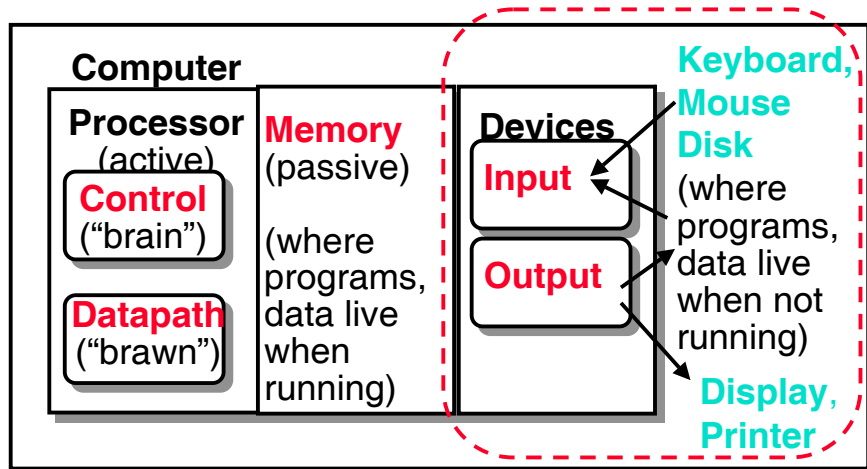
Overview

- Parallel Interfacing
- Serial Interfacing
 - UART
 - RS232

ELEC2041 lec29-io-examples.2

Saeid Nooshabadi

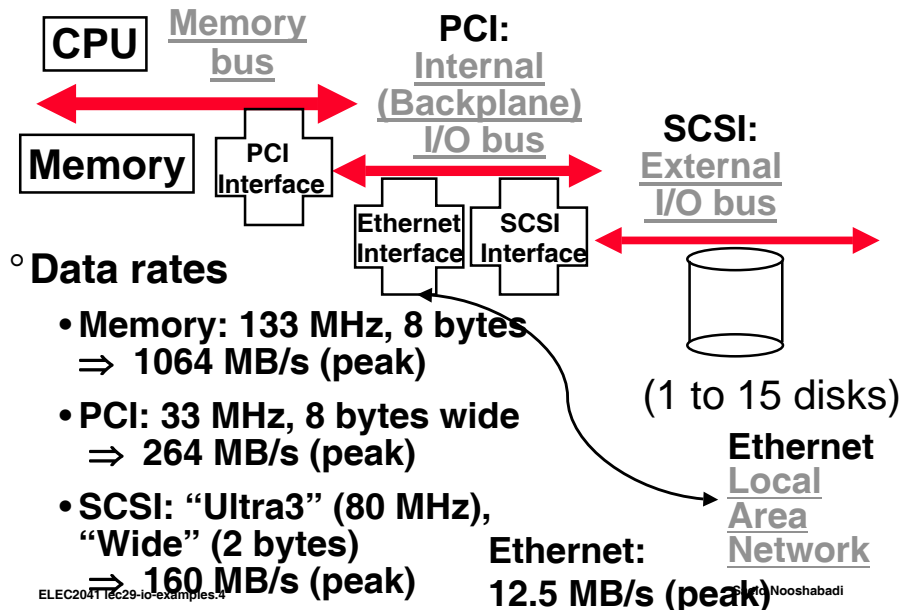
Anatomy: 5 components of any Computer



ELEC2041 lec29-io-examples.3

Saeid Nooshabadi

Review: Buses in a PC: Connect a few devices

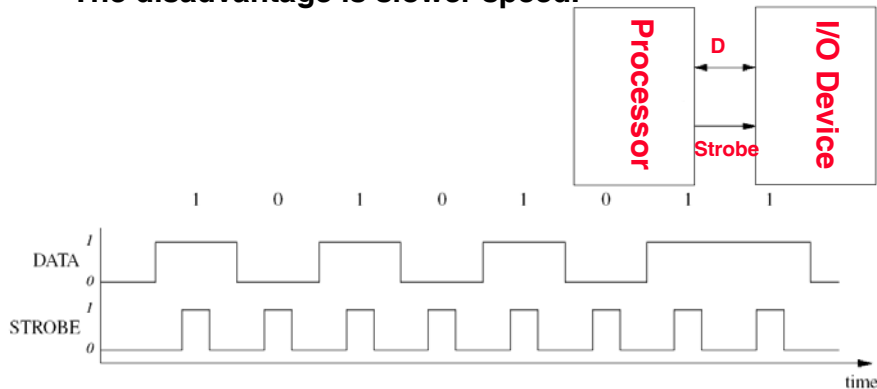


ELEC2041 lec29-io-examples.4

Saeid Nooshabadi

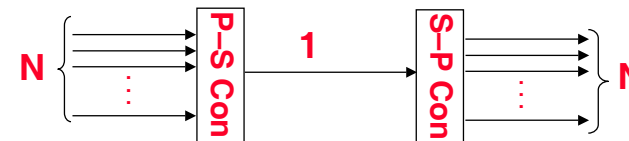
Serial Interfacing

- In serial I/O, the data bits are sent one at a time across a single line.
 - The advantage of serial I/O is lower cost (in terms of the number of wires connecting the microcomputer to peripheral device)
 - The disadvantage is slower speed.



Parallel ↔ Serial Interfacing

- Since communication within a microprocessor takes place over the system bus in parallel form, there is obviously a need for parallel-to serial (and serial-to parallel) conversion when interfacing to serial devices.

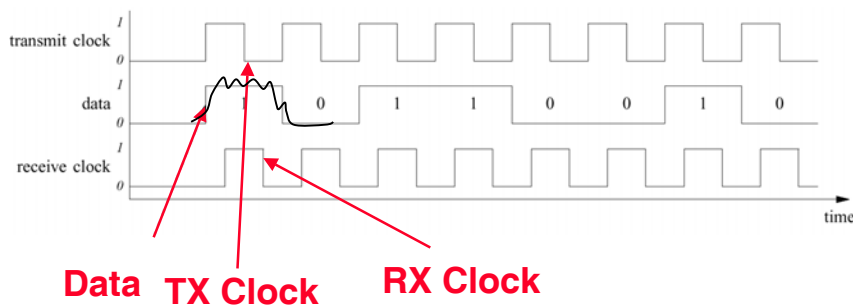


ELEC2041 lec29-io-examples.10

Saeid Nooshabadi

Asynchronous Serial Communication

- Used in character oriented data transmission between a microprocessor and an external device
 - Transmitter and Receiver each has its own clock running at the same frequency
 - How to synchronize two clocks so to sample in the middle of the data?



ELEC2041 lec29-io-examples.11

Saeid Nooshabadi

Making Asyn. Transmission Work

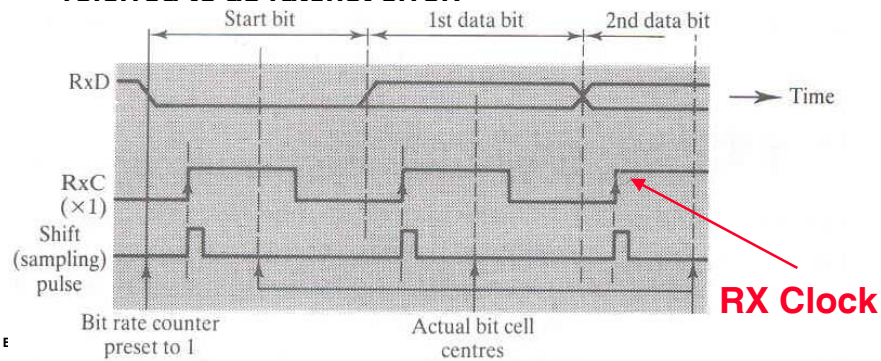
- Receiver Synchronisation:
 - The transmission of first bit should starts with a transition on the data line (1→0)
 - send an extra 'start' bit (= 0) before sending the 8-bit data,
 - data line is always set back to 1 at the end.
 - 1 → 0 transition always occurs at the start of each transmission.
 - the receive clock now samples 9 bits (start + 8 data bits),
 - the gap (idle time) between successive groups of 9 bits can change
 - Character wide synchronisation (Asynchronous)

ELEC2041 lec29-io-examples.12

Saeid Nooshabadi

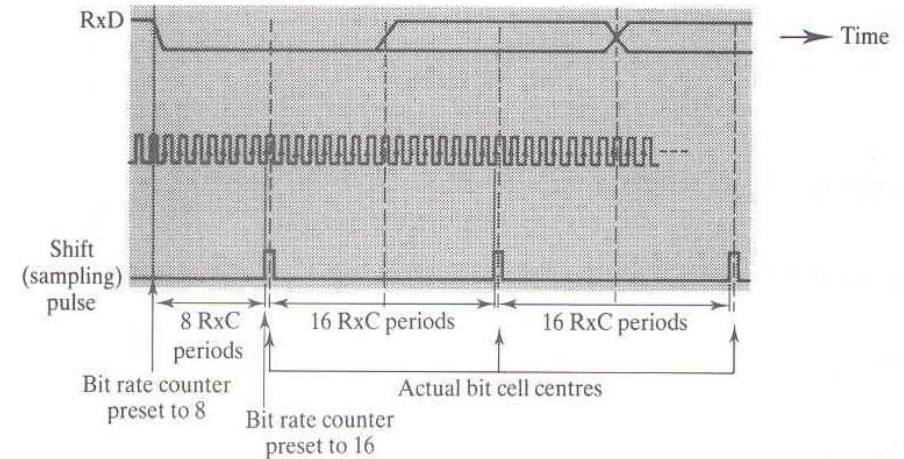
Receiver Clock Synchronisation Issues

- The receiver clock can be made equal to the baud rate
- clock must be very accurate in order to sample the incoming bit stream in the centre of its cycle.
- The sample point needs to be very close to the centre of the bit cell for reliable data recovery.
- The actual variation from the centre on the bit cell is referred to as ratchet error.



Improving Receiver Clock Synchronisation

- If the clock is made 16 times the baud rate, then the ratchet error can be relaxed from $\pm 1\%$ to $\pm 5\%$
- Ratchet relaxes to $\pm 25\%$ for 64 times the baud rate).



Parallel \leftrightarrow Serial Conversion

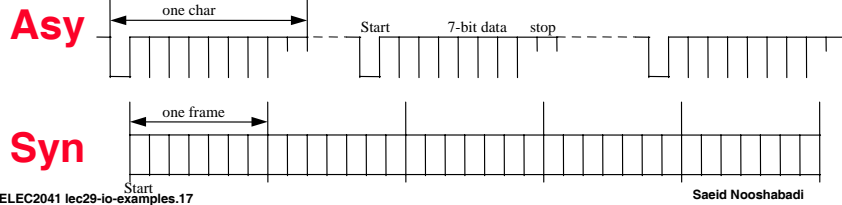
- Asynchronous data transmission uses a special device called Universal Asynchronous Receiver Transmitter (UART).
- UART is used to simultaneously transmit and receive serial data
- performs the appropriate parallel/serial conversions and inserting or checking the extra bits used to keep the serial data synchronised.
- UART typically configured as 2-4 I/O addresses: input/output status port(s), and output/input data port(s).
- Bytes sent as 8-bit parallel data to the output data address by the computer are converted into a standard-format serial bit stream for transmission by a transmitter inside the UART
- Similarly, an incoming serial bit stream is detected by a receiver inside the UART and converted into parallel data that can be read by the computer from the UART's input data address.

Full Duplex VS Half Duplex Data Transmission

- Simultaneous conversion of an incoming and an outgoing serial data stream is called full duplex
 - It requires two data carriers (TxD, and RxD)
 - Implemented with three wires: one for the outgoing stream (TxD), one for the incoming stream (RxD), and the third for a common ground line.
 - The UART does provide for standard full duplex handshaking conventions.
- Half duplex allows two-way communications, hence the name duplex, but only one direction is active at a time.

Synchronous Serial Data Transmission

- In **Asynchronous** data transmission TX and RX clocks are unsynchronised
 - Inefficient (for each 7 bits we send 3 – 4 extra bits)
 - Synchronisation across characters
- In **Synchronous** Data Transmission TX and RX clocks are synchronised
 - A common shared clock, (I²C), or clocking information embedded in the data stream (USB, Ethernet)
 - Fast (many bytes send before a re-synchronisation)
 - Synchronisation across frames vs characters



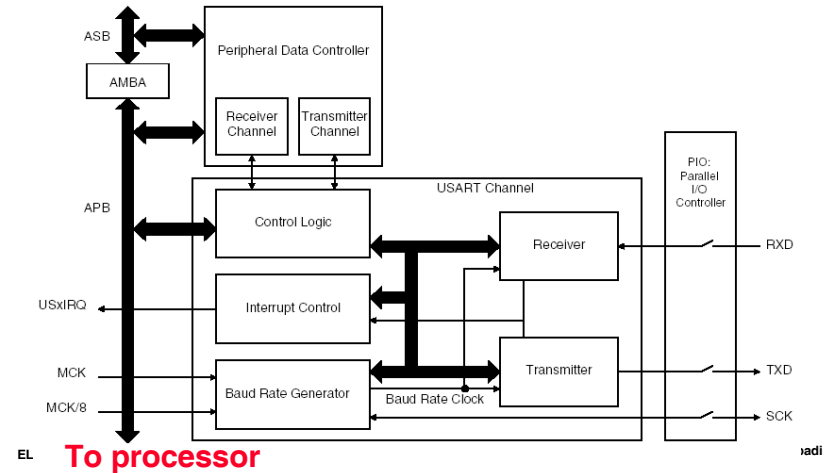
ELEC2041 lec29-io-examples.17

Saeid Nooshabadi

Serial Data Channels on AT91 on DSLMU Board

- Two Universal Synchronous Asynchronous Receiver Transmitter (USART)

- Programmable Baud rate
- Can generate interrupts



EL

iadi

DSLUMU/KOMODO Serial I/Os

- DSLMU Serial Port 1: memory-mapped terminal (Connected to the PC for program download and debugging)
 - Read from PC Keyboard (**receiver**); 2 device regs
 - Writes to PC terminal (**transmitter**); 2 device regs

| | | |
|-------------------------------|------------------|---------------|
| Receiver Status 0x10000014 | Unused (00...00) | Ready |
| Receiver Data 0x10000010 | Unused (00...00) | Received Byte |

| | | |
|----------------------------------|------------------|------------------|
| Transmitter Status 0x10000014 | Unused (00...00) | Ready |
| Transmitter Data 0x10000010 | Unused | Transmitted Byte |

ELEC2041 lec29-io-examples.19

Saeid Nooshabadi

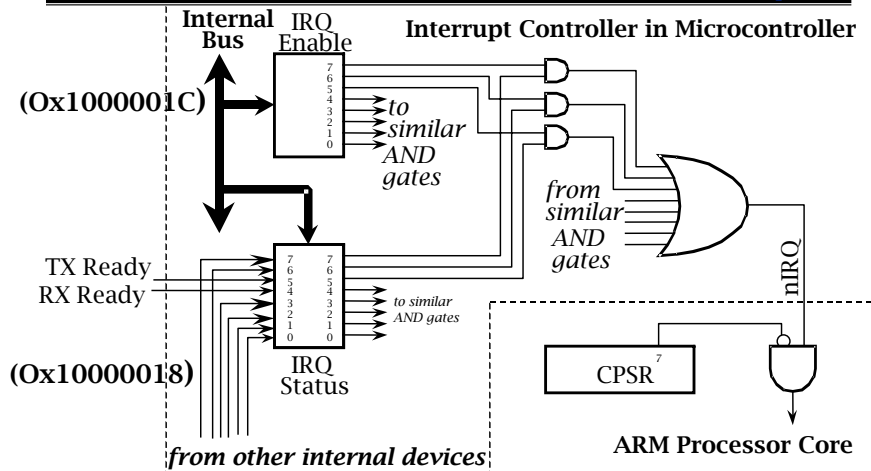
DSLUMU/Komodo Serial I/Os

- **Status register rightmost bit (0): Ready**
 - Receiver: Ready==1 means character in Data Register not yet been read (or ready to be read);
1 ⇒ 0 when data is read from Data Reg
 - Transmitter: Ready==1 means transmitter is ready to accept a new character;
0 ⇒ Transmitter still busy writing last char
- **Data register rightmost byte has data**
 - Receiver: last char from keyboard; rest = 0
 - Transmitter: when write rightmost byte, writes char to display

ELEC2041 lec29-io-examples.20

Saeid Nooshabadi

DSL/MU/KOMODO Serial I/Os Interrupts



- **IRQ Enable** : Enables individual interrupts
- **IRQ Status**: Indicates raising interrupt.
- **When a char is received or sent an interrupt is raised**

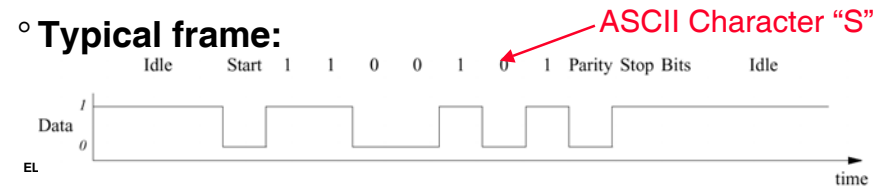
ELEC2041 lec29-io-examples.21

Saeid Nooshabadi

Asyn. Serial Communication Standard (RS232C)

- **Standard for communication of ASCII-coded character data between devices such as data computers and modems**
 - Low speed and cheap
- **Standard definition:**
 - The voltages used to represent 0 and 1 (Electrical)
 - The rate at which data is sent.
 - The format of the data sent.
 - The connectors to be used (physical and mechanical)
 - Extra control signals that may be used.
- **Typical data rate ((baud rate) are: 75, 300, 1200, 2400, 9600, 19200 and 115,000 bits/sec**

- **Typical frame:**



RS232C Definitions

- **The parity bit:**
 - is used as an error check.
 - The total number of '1's in the character+parity is made either odd (odd parity) or even (even parity).
 - Any single-bit error makes the parity bit appear wrong.
- **The stop bit(s):**
 - exist to allow for the case where one frame is transmitted immediately after another.
 - The stop bits, which are always 1, ensure the next start bit's 1 → 0 transition. (1, 1 1/2 or 2 bits)
- **Voltage values:**
 - >±5 should be used (Normally >±13 used)
 - +5 represents logic low (space) and -5 logic high (mark)
- **Physical characteristic:**
 - 25 way connector, (9 way is more popular now)

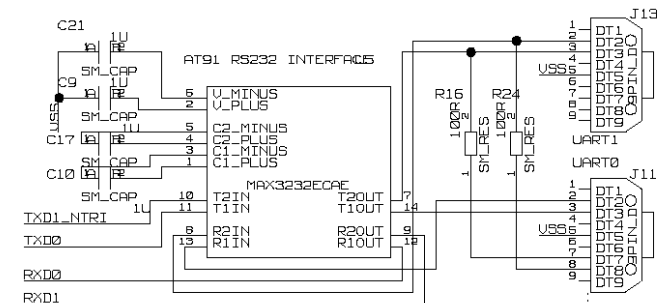
ELEC2041 lec29-io-examples.23

Saeid Nooshabadi

From UART to RS232-C

- **The UART is responsible for certain parts in RS232-C standard specifications:**
 - framing and transmitting TX data
 - receiving and extracting the RX data
 - baud rate generation
- **The electrical signaling is handled by a driver**
 - logic inversion and voltage translation

R232 Interface in DSLMU



ELEC2041 lec29-io-examples.24

Non Standard RS-232 Standard

- **RS-232 has earned the distinction of being the most non-standard standard in electronics!**
 - in general, two RS-232 devices, when connected together, won't work.
- **RS-232 was designed for connecting DTEs ("data terminal equipment") (like PC) to DCEs ("data communication equipment") (like modem).**
- **A DTE has a male and a DCE a female connector**
 - Corresponding pins in DTE connector connect to corresponding pins in DCE connector.
- **The IBM PC looks like a DTE with a male connector**
- **The DSLUM board also looks like a DTE with a male connector**
- **How to connect PC to DSLMU?**
 - Use "null modem"; cable that crosses TxD and RxD wires.

Reading Material

- **Reading Material:**
 - <http://www.beyondlogic.org/serial/serial.htm>
 - <http://www.sangoma.com/signal.htm>
 - **Hardware Reference Manual on CD-ROM**

ELEC2041 lec29-io-examples.26

Saeid Nooshabadi

"And In Conclusion"

- **Parallel Interfacing**
 - Fast but expensive
- **Serial Interfacing**
 - Slow but inexpensive
- **Synchronous Serial Interfacing**
 - Fast and more efficient but requires clock synchronisation
- **Asynchronous Serial Interfacing**
 - Slower and less efficient but does not require clock synchronisation
- **RS232 Standard**
 - The most widely used serial communication standard for communication between DTE and DCE devices

ELEC2041 lec29-io-examples.27

Saeid Nooshabadi