Device Drivers – LCD Display

Task Plan

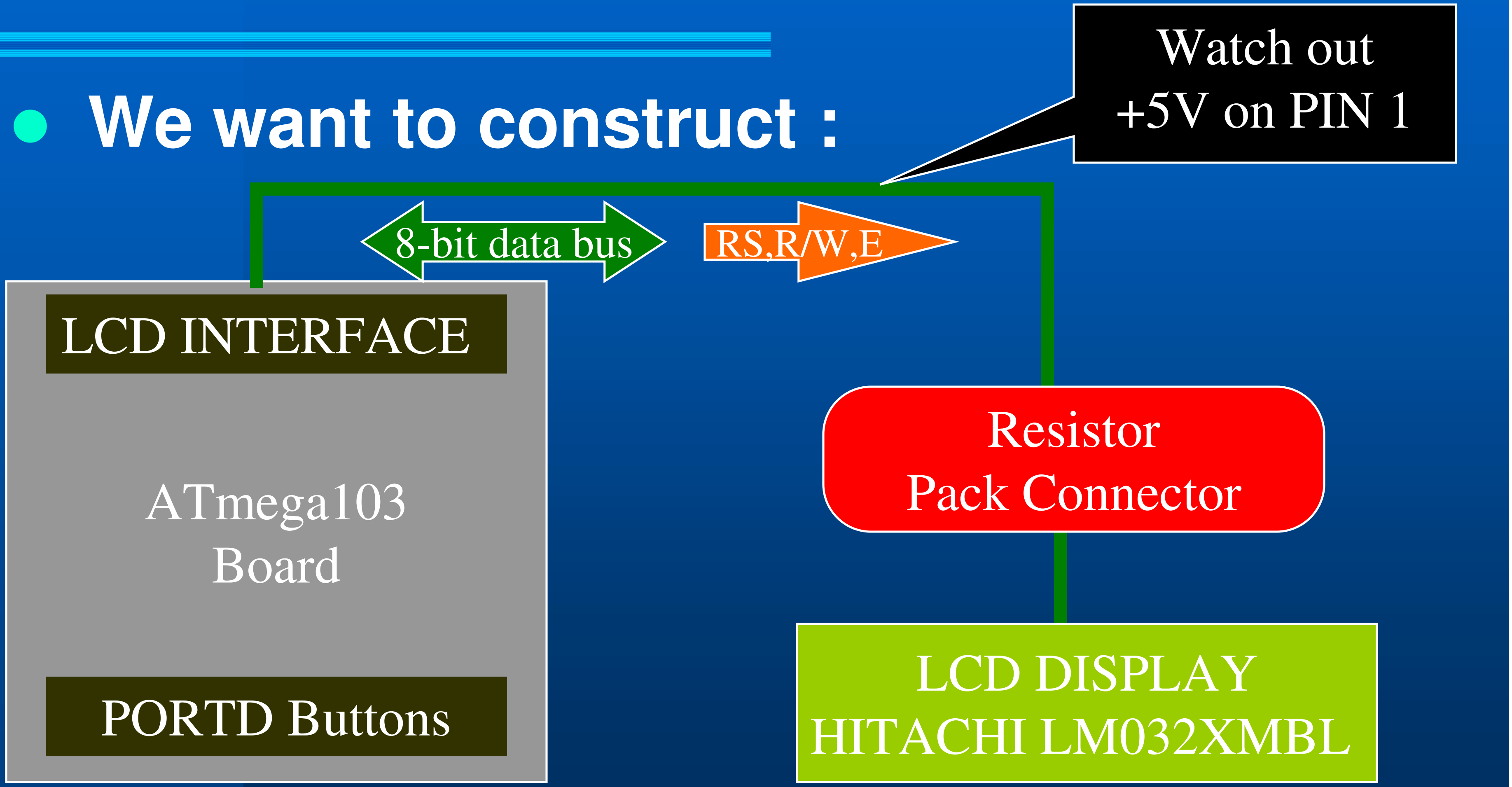
(1) You will create a device driver for a Liquid Crystal Display (LCD).

(2) You will learn some useful assembly commands.

***Lesson from last time*: Never connect the ATMEL board to external devices without the resistor pack connectors !!!!!**

Conceptual Design

- **We want to construct :**



Watch out
+5V on PIN 1

8-bit data bus

RS,R/W,E

LCD INTERFACE

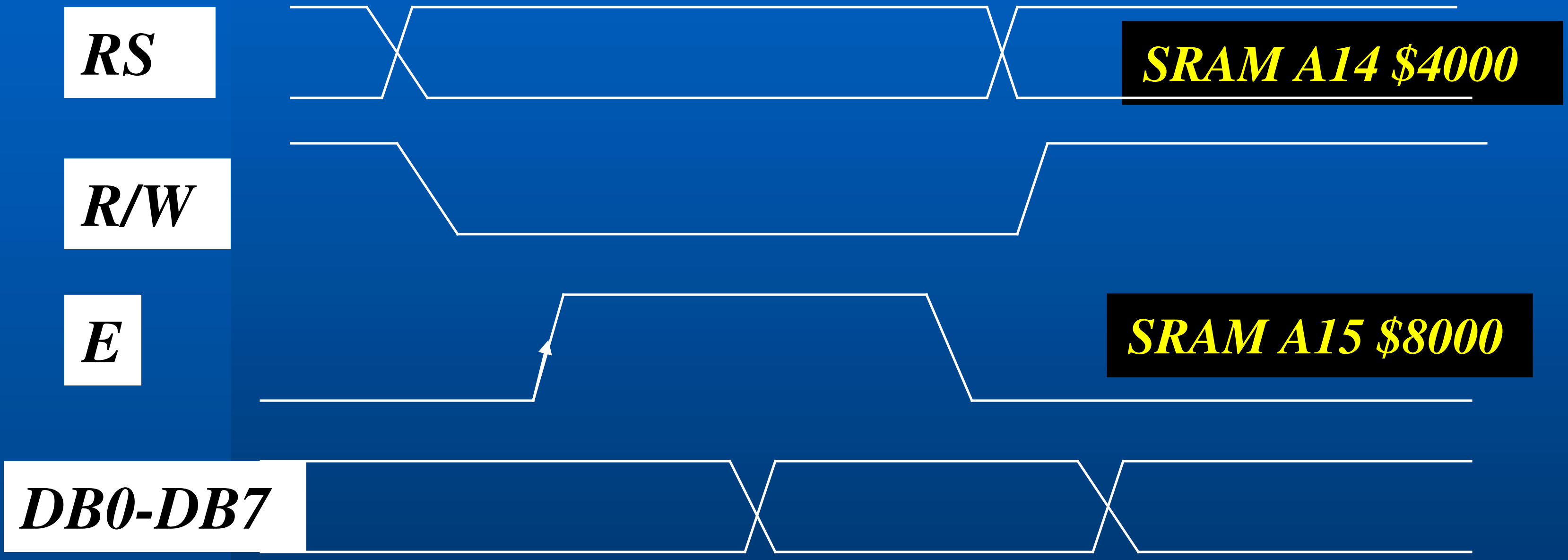
ATmega103
Board

PORTD Buttons

Resistor
Pack Connector

LCD DISPLAY
HITACHI LM032XMBL

The Hitachi LM032XMBL LCD



SRAM A14 $4000

SRAM A15 $8000

To write commands use: $8000
To write data use: $C000
To read the Busy flag use: $8000

Useful things to know

The LCD board is viewed as an external SRAM
From the ATMEL STK300+ board.

The External SRAM option must be enabled:

```
; ******* Sleep Mode And SRAM  *******
;                     ; tell it we want read and write activity on RE WR
ldi r16, $C0          ; Idle Mode - SE bit in MCUCR not set
out MCUCR, r16        ; External SRAM Enable Wait State Enabled
```

PORTA is used for address higher bits and data
PORTC is used for the address lower bits.

Getting Started

(1) Download LCD1.asm from the Lab page.

(2) In the file you will find routines to initialize the LCD display, routines that clear the display and routines that print messages.

(3) Hack them to create an LCD interface of your own and familiarize yourself with them

Assembly III: Main Program

```
        CLR r23
        SBR r23, $1
Main:
; Do not execute the initialization every time;
        SBRC r23, 0
        rcall Idisp
        SBRC r23, 0
        CBR r23, $1
;
        rcall Mess1Out
        rcall BigDel
        rcall CLRDIS
        rcall Mess2Out
        rcall BigDel
        rcall CLRDIS
        rjmp Main
```

Tables, Messages, ASCII, Bytes

```
Mess1:
.db 'C','o','s','t','a','s',' ','F','o','u','d','a','s'
Mess1Out:
        LDI ZH, HIGH(2*Mess1)
        LDI ZL, LOW(2*Mess1)
        LDI r18, 13
Mess1More:
        LPM
        MOV r17, r0
        sts $C000, r17
        rcall busylcd
        DEC r18
        BREQ Mess1End
        ADIW ZL, $01
        ;OUT PORTB, ZL
        RJMP Mess1More
Mess1End:   ret
```

Use 16 bit counters for delays

```
DEL49ms:
        LDI XH, HIGH(65535)
        LDI XL, LOW (65535)
COUNT3:
        SBIW XL, 1
        BRNE COUNT3
        RET
```

The Skip assembly commands

```
; A routine the probes the display BUSY bit
;

busylcd:
            rcall Del49ms
            lds r16, $8000         ; read the lcd BUSY bit
            sbrs r16, 7            ;skip next if busy bit  7 is set
            ret                    ;return if clear
            rjmp busylcd
;
```

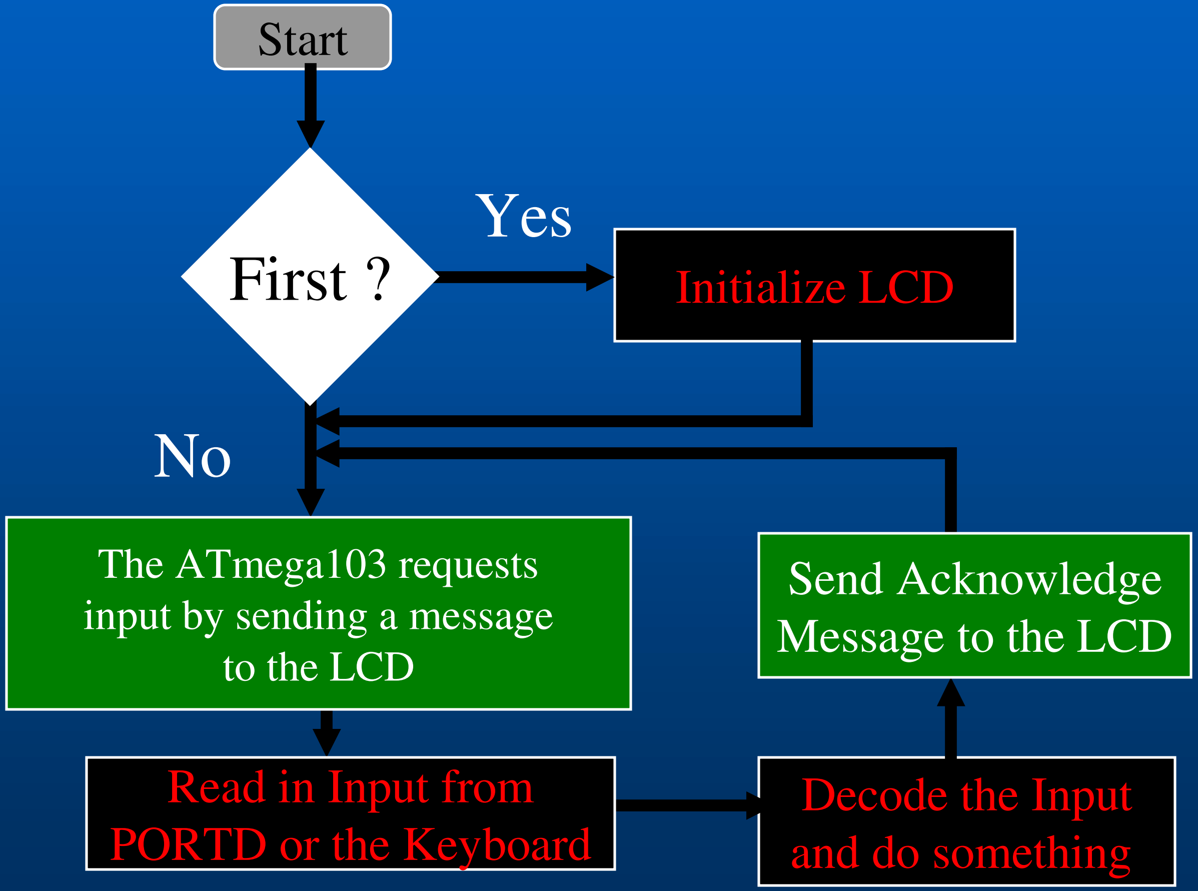
You can skip with sbrs, sbrc for registers and with sbis, sbic for I/O Ports

You may clear the display using :

```
; This clears the display so we can start all over again
;
CLRDIS:
            ldi r16,$01              ; Clear Display send cursor
            sts $8000,r16            ; to the most left position
            rcall busylcd
            ret
```

In the Hitachi read and white book you will find lots of commands for the LCD display.

Exercise: A User Interface


Start
First ?
Yes
Initialize LCD
No
The ATmega103 requests input by sending a message to the LCD
Send Acknowledge Message to the LCD
Read in Input from PORTD or the Keyboard
Decode the Input and do something

Project: Make a Signal Generator

Your Signal Generator should be programmable.

You should be able to use the keyboard and the LCD display to change the:

(1) Frequency
(2) Amplitude
(3) Offset
(4) Pulse shape (Square, Sinusoidal, Triangle..)

You will crate the analog pulses by using an Digital to Analog Converter (DAC)

Other ideas for a project

(1) Alarm Clock
(2) Home Security System
(3) Digital Thermometer
(4) Calculator
(5) Cycle Computer
(6) Distance ranging device
(7) Programming Language (parser/Interpreter)
(8) Guitar tuner
(9) Elevator Control
(10) I2C Master and Slave interfaces