Lecture #1 Outline

• Welcome Back
• AVR Hardware
  – Pins and Ports
• AVR Programming
  – AVR Memory and Addressing Modes
  – AVR Instruction Set
  – Timers
• Project Ideas
Survey: Have you…

- Subscribed to the E-mail list?
- Started on Lab #0 Yet?
- Started on Lab #1 Yet?
- Got access to the EE281 lab yet?
- Got your account set up on the lab machines?
- Read over the AVR Instruction Set?
- Played with AVR Studio?
- Simulated blink.asm?
Lab Kits: STK500 for all

- Get these items after class
  - Atmel AVR STK500
    - The AT90S8515 processor
    - Configurable options:
      - Port connections (LEDs and switches)
      - Supply Voltage (*be careful)
      - Processor frequency
  - Atmel Web Site CD (everything and more)
  - Power Transformer
  - Databook (*while they last)
Software Notes

• AVR Studio 4.0
  – The only software you need for the STK500
    • Works as editor for Assembly and C
    • Has built-in AVR assembler
    • Has built-in AVR simulator
    • Has built-in STK500 programmer

• PonyProg2000
  – Don’t need it for the STK500
  – But…it’s perfect for programming your final project!
AVR AT90S8515 Pinout

- General Purpose Ports
  - PORTA
  - PORTB
  - PORTC
  - PORTD
  - (Special Functions)
- Special Purpose Pins
  - Crystal (XTAL1/XTAL2)
  - RESET
  - ICP, OLE, OC1B
- Power (VCC/GND)
• 32 Registers (R0-R31)
• 4K Prog ROM
• 512 bytes RAM
• 512 bytes EEPROM
• 32 I/O lines
• 13 Interrupts
• Lots of fun built-in peripherals
AVR Memory Spaces

- **Program Flash**
  - Vectors, Code, and (Unchangeable) Constant Data
- **Working Registers**
  - Includes X, Y, and Z registers.
- **I/O Register Space**
  - Includes “named” registers
- **SRAM – Data Space**
  - Runtime Variables and Data
  - Stack space
- **EEPROM space**
  - For non-volatile but alterable data
# AVR Data Memory Map: Part 1

## Register File

<table>
<thead>
<tr>
<th>Register</th>
<th>Data Address Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0</td>
<td>$0000</td>
</tr>
<tr>
<td>R1</td>
<td>$0001</td>
</tr>
<tr>
<td>R2</td>
<td>$0002</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>R29</td>
<td>$001D</td>
</tr>
<tr>
<td>R30</td>
<td>$001E</td>
</tr>
<tr>
<td>R31</td>
<td>$001F</td>
</tr>
</tbody>
</table>

## I/O Registers

<table>
<thead>
<tr>
<th>I/O Register</th>
<th>Data Address Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>$00</td>
<td>$0020</td>
</tr>
<tr>
<td>$01</td>
<td>$0021</td>
</tr>
<tr>
<td>$02</td>
<td>$0022</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$3D</td>
<td>$005D</td>
</tr>
<tr>
<td>$3E</td>
<td>$005E</td>
</tr>
<tr>
<td>$3F</td>
<td>$005F</td>
</tr>
</tbody>
</table>
## Internal SRAM

<table>
<thead>
<tr>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0060</td>
</tr>
<tr>
<td>$0061</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>$015E/$025E</td>
</tr>
<tr>
<td>$015F/$025F</td>
</tr>
</tbody>
</table>

## External SRAM

<table>
<thead>
<tr>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0160/$0260</td>
</tr>
<tr>
<td>$0161/$0261</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>$FFFFE</td>
</tr>
<tr>
<td>$FFFFF</td>
</tr>
</tbody>
</table>
AVR I/O Memory Map

- Peripherals and Ports are I/O mapped
- Use IN/OUT instructions to access
- Excerpt from I/O map showing ports:

<table>
<thead>
<tr>
<th>$1D ($3D)</th>
<th>EEDR</th>
<th>-</th>
<th>-</th>
<th>-</th>
<th>-</th>
<th>EEMWE</th>
<th>EEWE</th>
<th>EERE</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1C ($3C)</td>
<td>EECCR</td>
<td>PORTA7</td>
<td>PORTA6</td>
<td>PORTA5</td>
<td>PORTA4</td>
<td>PORTA3</td>
<td>PORTA2</td>
<td>PORTA1</td>
</tr>
<tr>
<td>$1B ($3B)</td>
<td>PORTA</td>
<td>PORTA7</td>
<td>PORTA6</td>
<td>PORTA5</td>
<td>PORTA4</td>
<td>PORTA3</td>
<td>PORTA2</td>
<td>PORTA1</td>
</tr>
<tr>
<td>$1A ($3A)</td>
<td>DDRA</td>
<td>DDA7</td>
<td>DDA6</td>
<td>DDA5</td>
<td>DDA4</td>
<td>DDA3</td>
<td>DDA2</td>
<td>DDA1</td>
</tr>
<tr>
<td>$19 ($39)</td>
<td>PINA</td>
<td>PINA7</td>
<td>PINA6</td>
<td>PINA5</td>
<td>PINA4</td>
<td>PINA3</td>
<td>PINA2</td>
<td>PINA1</td>
</tr>
<tr>
<td>$18 ($38)</td>
<td>PORTB</td>
<td>PORTB7</td>
<td>PORTB6</td>
<td>PORTB5</td>
<td>PORTB4</td>
<td>PORTB3</td>
<td>PORTB2</td>
<td>PORTB1</td>
</tr>
<tr>
<td>$17 ($37)</td>
<td>DDRB</td>
<td>DDB7</td>
<td>DDB6</td>
<td>DDB5</td>
<td>DDB4</td>
<td>DDB3</td>
<td>DDB2</td>
<td>DDB1</td>
</tr>
<tr>
<td>$16 ($36)</td>
<td>PINB</td>
<td>PINB7</td>
<td>PINB6</td>
<td>PINB5</td>
<td>PINB4</td>
<td>PINB3</td>
<td>PINB2</td>
<td>PINB1</td>
</tr>
<tr>
<td>$15 ($35)</td>
<td>PORTC</td>
<td>PORTC7</td>
<td>PORTC6</td>
<td>PORTC5</td>
<td>PORTC4</td>
<td>PORTC3</td>
<td>PORTC2</td>
<td>PORTC1</td>
</tr>
<tr>
<td>$14 ($34)</td>
<td>DDRC</td>
<td>DDC7</td>
<td>DDC6</td>
<td>DDC5</td>
<td>DDC4</td>
<td>DDC3</td>
<td>DDC2</td>
<td>DDC1</td>
</tr>
<tr>
<td>$13 ($33)</td>
<td>PINC</td>
<td>PINC7</td>
<td>PINC6</td>
<td>PINC5</td>
<td>PINC4</td>
<td>PINC3</td>
<td>PINC2</td>
<td>PINC1</td>
</tr>
<tr>
<td>$12 ($32)</td>
<td>PORTD</td>
<td>PORTD7</td>
<td>PORTD6</td>
<td>PORTD5</td>
<td>PORTD4</td>
<td>PORTD3</td>
<td>PORTD2</td>
<td>PORTD1</td>
</tr>
<tr>
<td>$11 ($31)</td>
<td>DDRD</td>
<td>DDD7</td>
<td>DDD6</td>
<td>DDD5</td>
<td>DDD4</td>
<td>DDD3</td>
<td>DDD2</td>
<td>DDD1</td>
</tr>
<tr>
<td>$10 ($30)</td>
<td>PIND</td>
<td>PIND7</td>
<td>PIND6</td>
<td>PIND5</td>
<td>PIND4</td>
<td>PIND3</td>
<td>PIND2</td>
<td>PIND1</td>
</tr>
<tr>
<td>$0F ($2F)</td>
<td>SPD</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$0E ($2E)</td>
<td>SPRSR</td>
<td>SPIF</td>
<td>WCOL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$0D ($2D)</td>
<td>SPCR</td>
<td>SDIE</td>
<td>SDE</td>
<td>SDDR</td>
<td>MSTR</td>
<td>CD0</td>
<td>CD1</td>
<td>CD2</td>
</tr>
</tbody>
</table>

September 27, 2002  Stanford University - EE281 Lecture #1
AVR Addressing Modes

- Register Direct, with 1 and 2 registers
- I/O Direct
- Data Direct
- Data Indirect
  - with pre-decrement
  - with post-increment
- Code Memory Addressing
Register Direct: 1 Register

Examples:
INC R16
CLR R22
EOR R0
Register Direct: 2 Registers

Examples:
ADD   R16, R17
CP    R22, R5
MOV   R0, R1
I/O Direct

Examples:
IN R16, PIND
OUT PORTC, R16
Data Direct

Examples:
STS 0x1000, R16
Data Indirect

Examples:

LD R16, Y
ST Z, R16
Data Indirect w/ Displacement

Examples:
LDD R16, Y+0x10
STD Z+0x20, R16
Data Indirect: Pre-Decrement

Examples:

\[ \text{LD} \quad R16, -Z \]
\[ \text{ST} \quad -Z, R16 \]
Data Indirect: Post-Increment

Examples:
LD  R16, Z+
ST  Z+, R16
Prog. Memory Addressing

Examples:

LPM

PROGRAM MEMORY

$000

$7FF/$FFF
Indirect Program Addressing

Examples:
IJMP
ICALL
Relative Program Addressing

Examples:
- RJMP
- RCALL

PROGRAM MEMORY

$000

$7FF/$FFF

PC

15 12 11 0

OP k
I/O Register $3F : SREG

• All Bits are R/W:
  – I – Global Interrupt Enable
  – T – Bit Copy Storage
  – H – Half Carry Flag
  – S – Sign Bit
  – V – Two’s Complement Overflow Flag
  – N – Negative Flag
  – Z – Zero Flag
  – C – Carry Flag
Instruction Examples: Add

- Math
  - Add
    - ADD Rd, Rr – Adds two registers
      - Rd <- Rd + Rr
    - ADC Rd, Rr – Add with Carry two registers
      - Rd <- Rd + Rr + C
    - ADIW Rd1, K – Add Immediate to Word
      - Rdh:Rdl <- Rdh:Rdl + K
Other Math and Logic

- Subtract
- Logical AND
- Logical OR
- Exclusive OR
- One’s Complement
- Two’s Complement
- Increment/Decrement
- Set/Clear Registers and Bits in Registers
Branch Instructions

- **RJMP/RCALL** – Relative Jmp (+/-k)
- **IJMP/ICALL** – Indirect Jmp (Z Reg)
- **RET/RETI** – Return from call/interrupt
- **CP** - Compare
- **SB** - Skip if Bit in Register or I/O is set/clear
- **BR** - Branch if condition is met
Data Transfer Instructions

- **MOV** – Move between registers
- **LD/LDI** – Load / Load Immediate
- **ST/STI** – Store / Store Immediate
- **LPM** – Load Program Memory
  - Hardwired to load R0 with (Z) in code.
- **IN/OUT** – In and Out Ports
- **PUSH/POP** – On and off stack
Bit and Bit Test Instructions

- **SBI/CBI** – Set / Clear Bit in register
- **LSL/LSR** – Logical Shift Left / Right
- **ROL/ROR** – Rotate Left / Right (thru Carry bit)
- **ASR** – Arithmetic Shift Right
- **SWAP** – Swap Nibbles
- **BST/BLD** – Bit Store / Load
- **BSET/BCLR** – Set / Clear Status Bits by number
- **SE*/CL* - Set / Clear Status Bits by name**
Other Instructions

- **NOP** – Do nothing for 1 cycle
- **SLEEP** – Sleep until reset or interrupted
- **WDR** – Watch Dog Reset
AVR Timer/Counter 0

- 8 Bit
- Wrap-Around Up Counter
- Interrupt on Overflow
AVR Timer/Counter 0 (cont’d)
AVR Timer/Counter 1

- 16 Bit
- Dual Comparators A,B (output captures)
- Up Counter
- Interrupt on:
  - Overflow
  - Compare A/B
  - Input Capture of external event on ICP pin.
- Can also act as an 8, 9 or 10 bit PWM Up-Down Counter.
Timer Control: I/O space

- Timer 0:
  - Control Register (TCCR0)
  - Timer/Counter0 (TCNT0)

- Timer 1:
  - Control Register A & B (TCCR1A/B)
  - Input Capture Register (ICR1)
  - Timer/Counter1 Output Compare Register A and B (OCR1A/B)
  - Timer/Counter1 (TCNT1)

- Timer Interrupt Registers (Mask and Flag Registers) are Common to Both Timers
AVR Timer/Counter Sources

- Shut Off
- CPU frequency divided by 1, 8, 64, 256, 1024
  - At 8MHz that’s: 1/8uS, 1uS, 8uS, 32uS, 128uS
- External Input (rising or falling).

<table>
<thead>
<tr>
<th>CS02</th>
<th>CS01</th>
<th>CS00</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Stop, the Timer/Counter0 is stopped.</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>CK</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>CK/8</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>CK/64</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>CK/256</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>CK/1024</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>External Pin T0, falling edge</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>External Pin T0, rising edge</td>
</tr>
</tbody>
</table>
Project Ideas?

• Automatic Timed Fish Feeder (extra credit :-(
• Laser Light Show Controller
• Animated Light Controller (stimulated by music?)
• Programmable Universal Remote Control
• Reactive Juggling Ball (accelerometers, FSR)
• Sports Car Performance Analyzer
• Very Small Web Server
• Communication Gadget
• Automated Remote Weather Station