EE281

Embedded System
Design Laboratory

Lecture #1 Outline

- Welcome Back
- AVR Hardware
  - Pins and Ports
- AVR Programming
  - AVR Memory and Addressing Modes
  - AVR Instruction Set
  - Timers
- Project Ideas

EE281

Embedded System
Design Laboratory

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?

EE281

Lab Kits: STK500 for all

Embedded System
Design Laboratory

- 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)

EE281

Embedded System
Design Laboratory

Software Notes

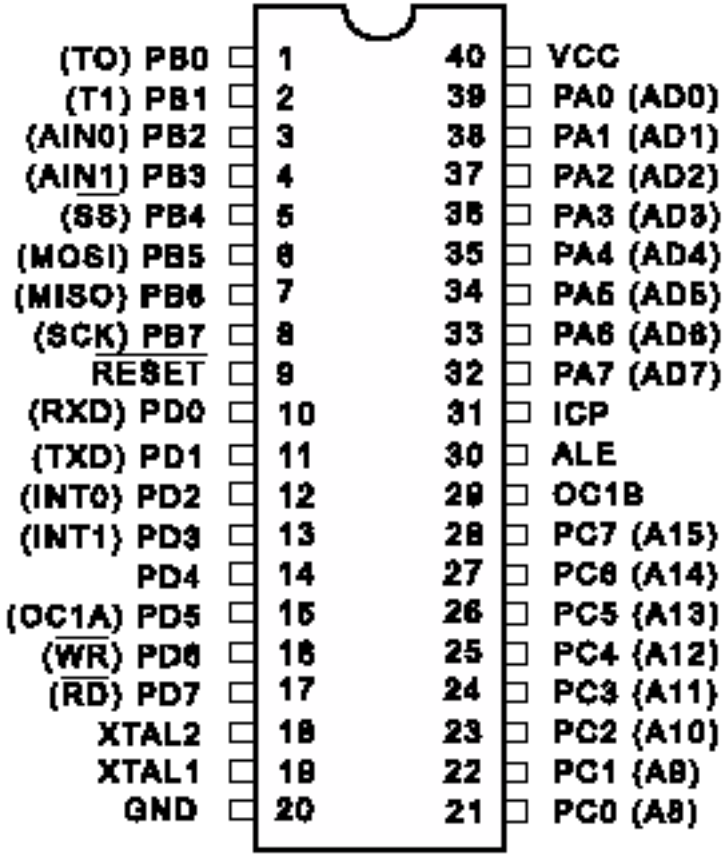
- AVR Studio 4.0
  - The only software you need for the STK500
    - Works as editor for Assembly and C
    - Has built-in AVR assembler
    - Has built-in AVR simulator
    - Has built-in STK500 programmer
- PonyProg2000
  - Don't need it for the STK500
  - But…it's perfect for programming your final project!

AVR AT90S8515 Pinout

EE281

Embedded System
Design Laboratory

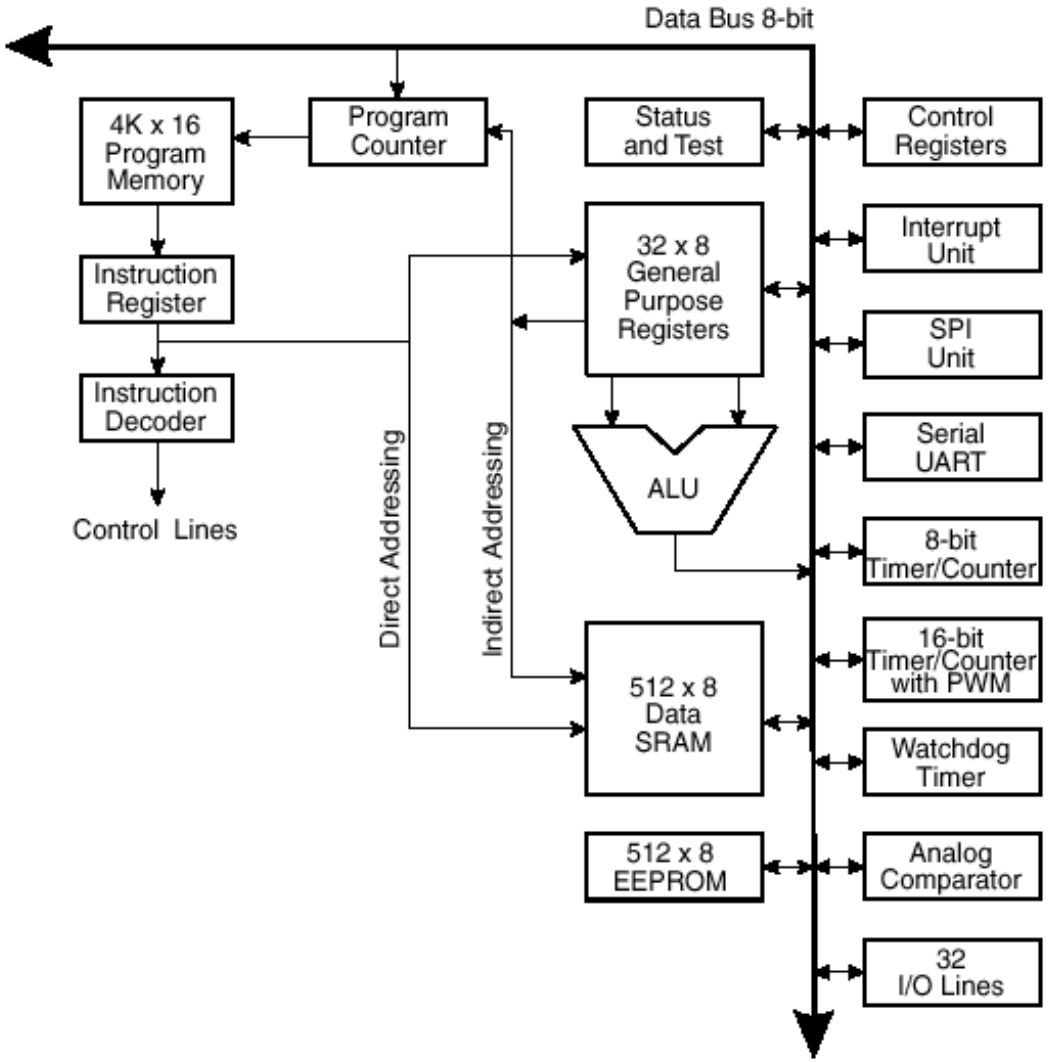
- General Purpose Ports
  - PORTA
  - PORTB
  - PORTC
  - PORTD
  - (Special Functions)
- Special Purpose Pins
  - Crystal (XTAL1/XTAL2)
  - RESET
  - ICP, OLE, OC1B
- Power (VCC/GND)



EE281 '8515 Functional Architecture

Embedded System
Design Laboratory

- 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



EE281

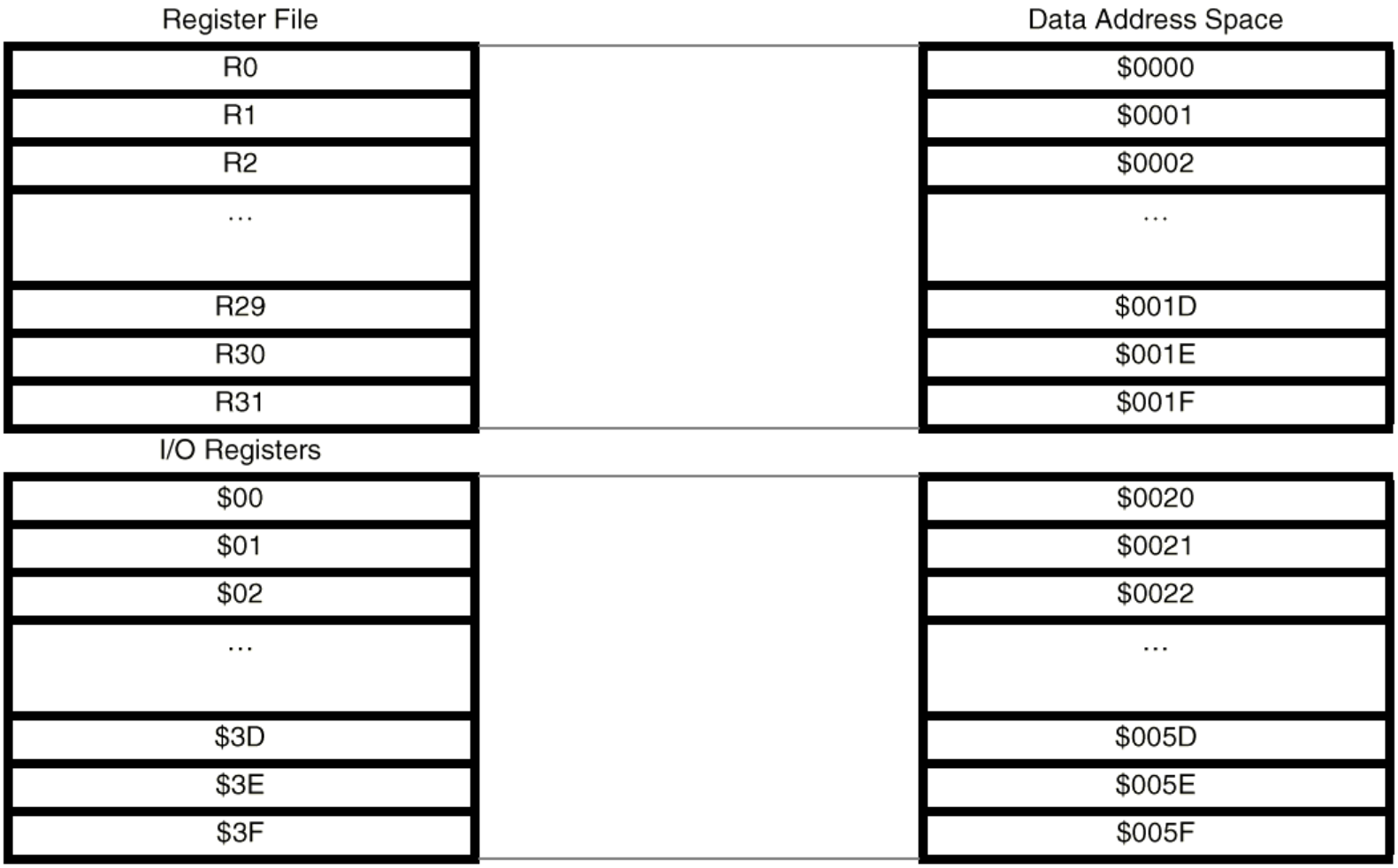
Embedded System
Design Laboratory

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

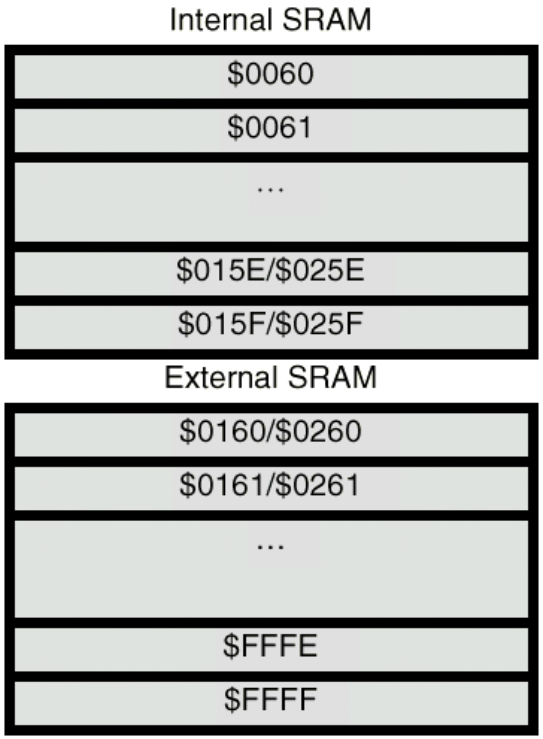
EE281 AVR Data Memory Map: Part 1

Embedded System Design Laboratory



EE281 AVR Data Memory Map: Part 2

Embedded System
Design Laboratory



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:

$1D ($3D)	EEDR	EEPROM Data Register								page 39
$1C ($3C)	EECR	-	-	-	-	-	EEMWE	EEWE	EERE	page 39
$1B ($3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	page 54
$1A ($3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	page 54
$19 ($39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	page 54
$18 ($38)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	page 56
$17 ($37)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	page 56
$16 ($36)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	page 56
$15 ($35)	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	page 61
$14 ($34)	DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	page 61
$13 ($33)	PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	page 61
$12 ($32)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	page 63
$11 ($31)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	page 63
$10 ($30)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	page 63
$0F ($2F)	SPDR	SPI Data Register								page 44
$0E ($2E)	SPSR	SPIF	WCOL	-	-	-	-	-	-	page 44
$0D ($2D)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	page 43

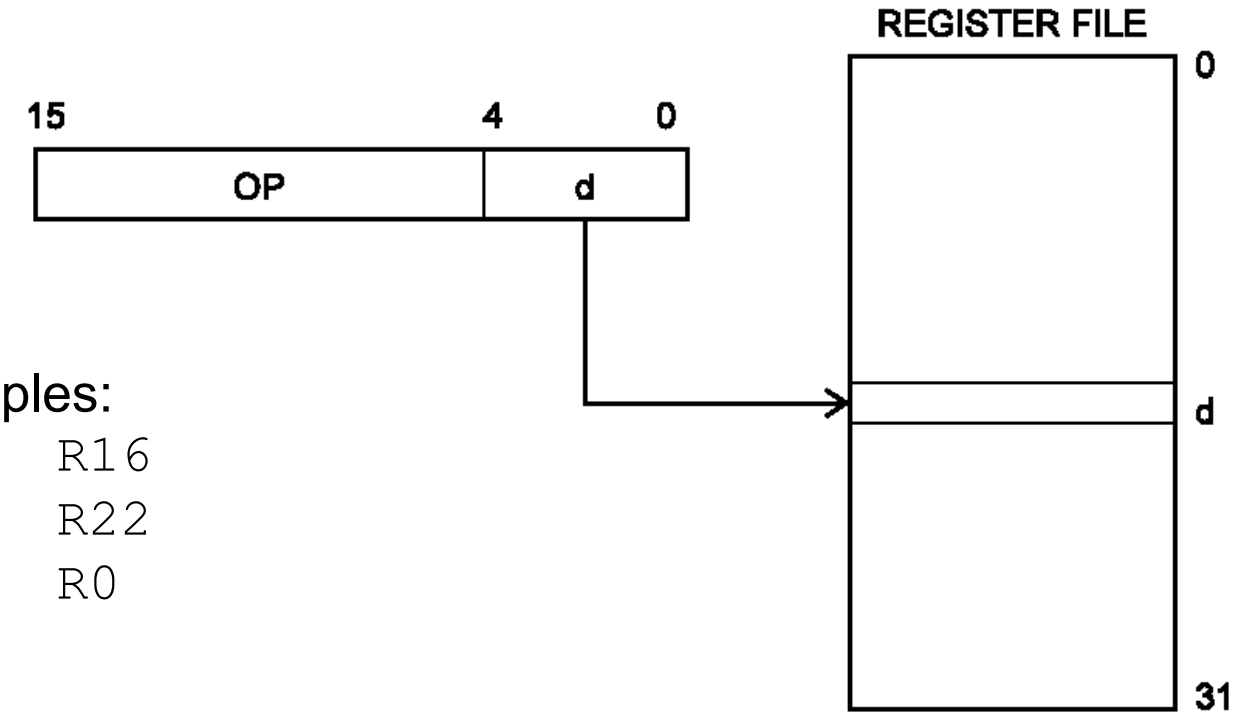
EE281

AVR Addressing Modes

Embedded System
Design Laboratory

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

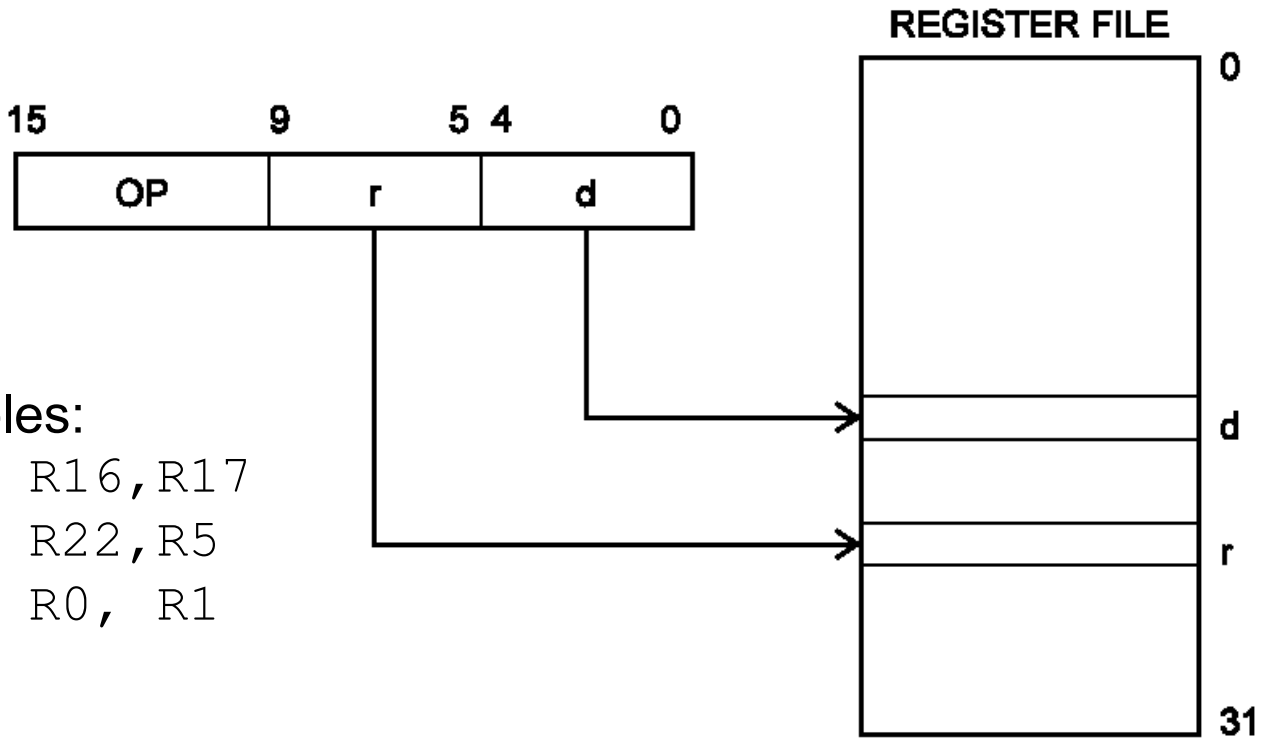
Register Direct: 1 Register



Examples:

```
INC    R16
CLR    R22
EOR    R0
```

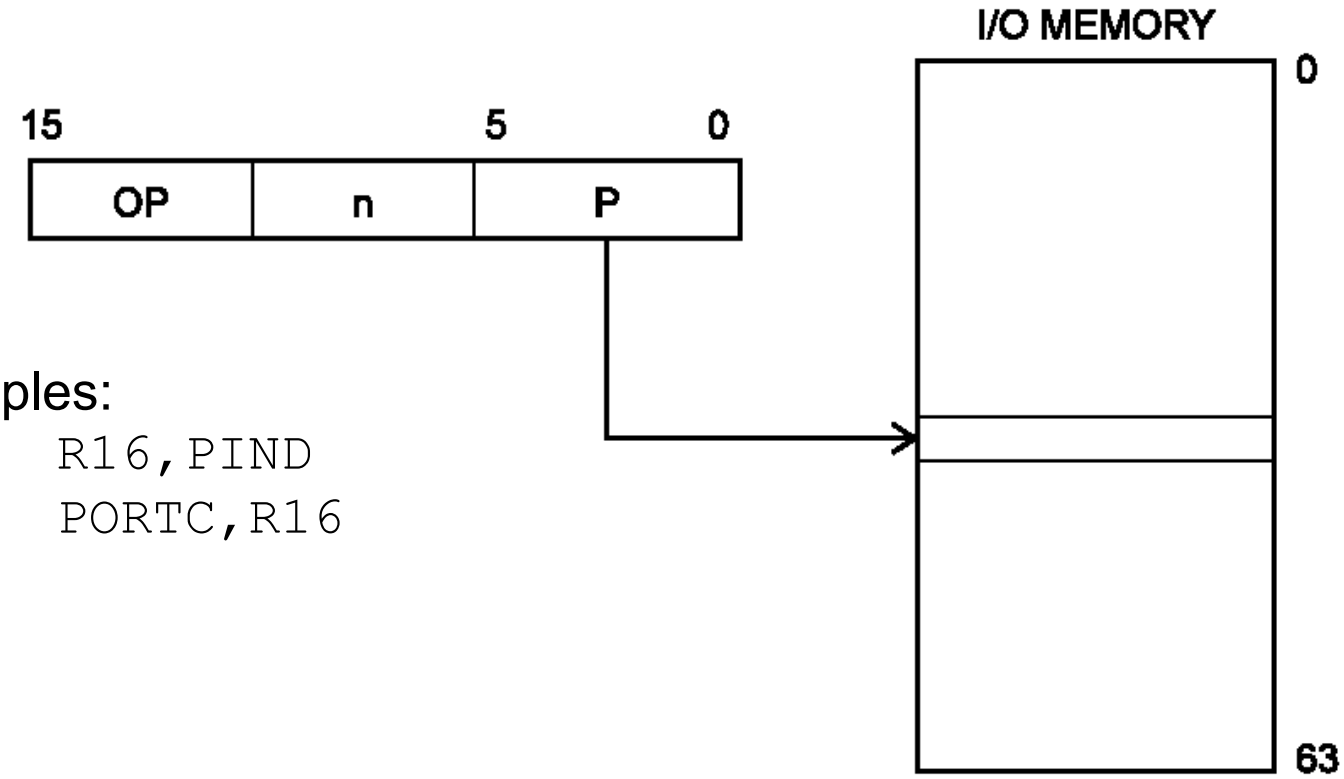
Register Direct: 2 Registers



Examples:

```
ADD    R16,R17
CP     R22,R5
MOV    R0, R1
```

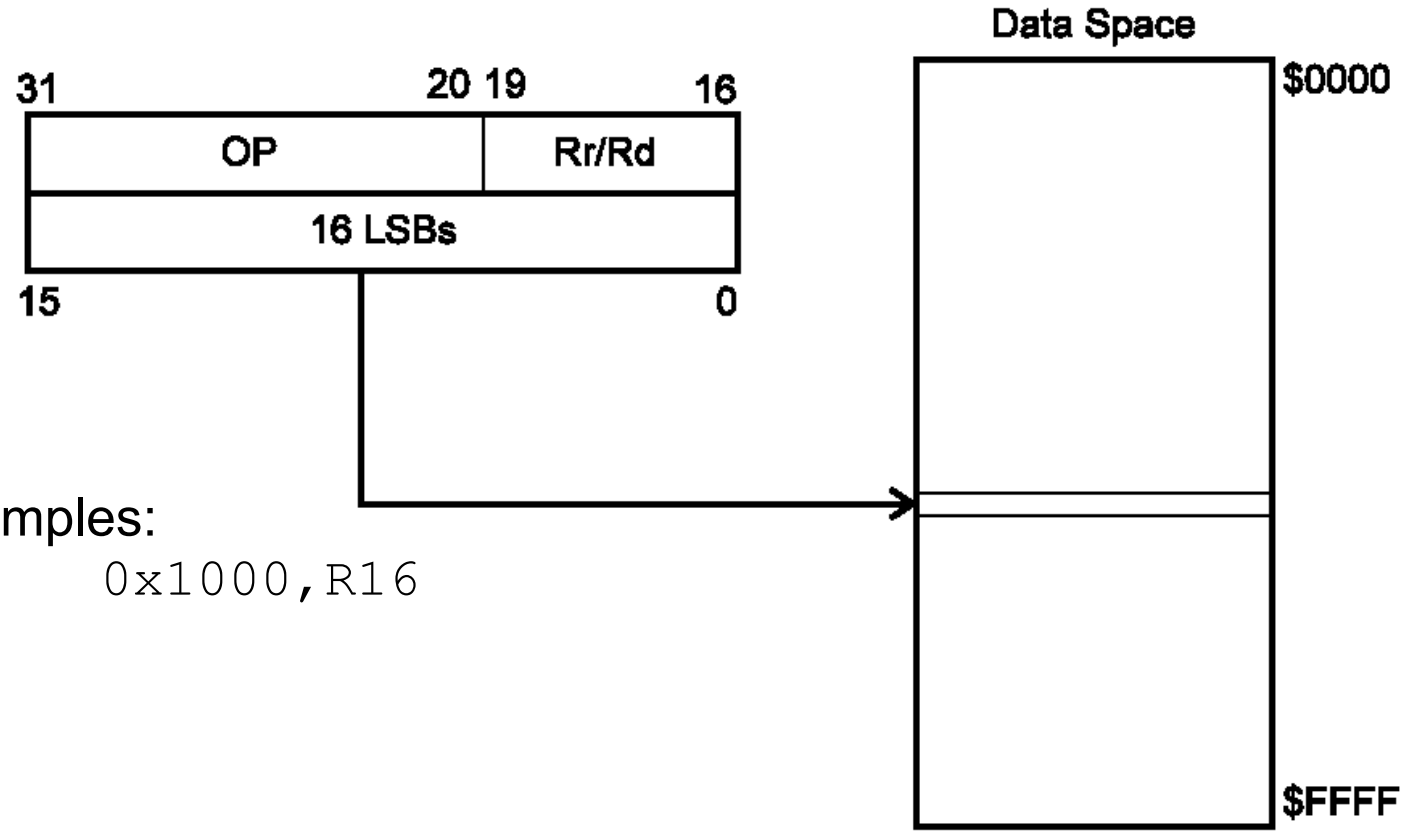
I/O Direct



Examples:

```
IN     R16,PIND
OUT    PORTC,R16
```

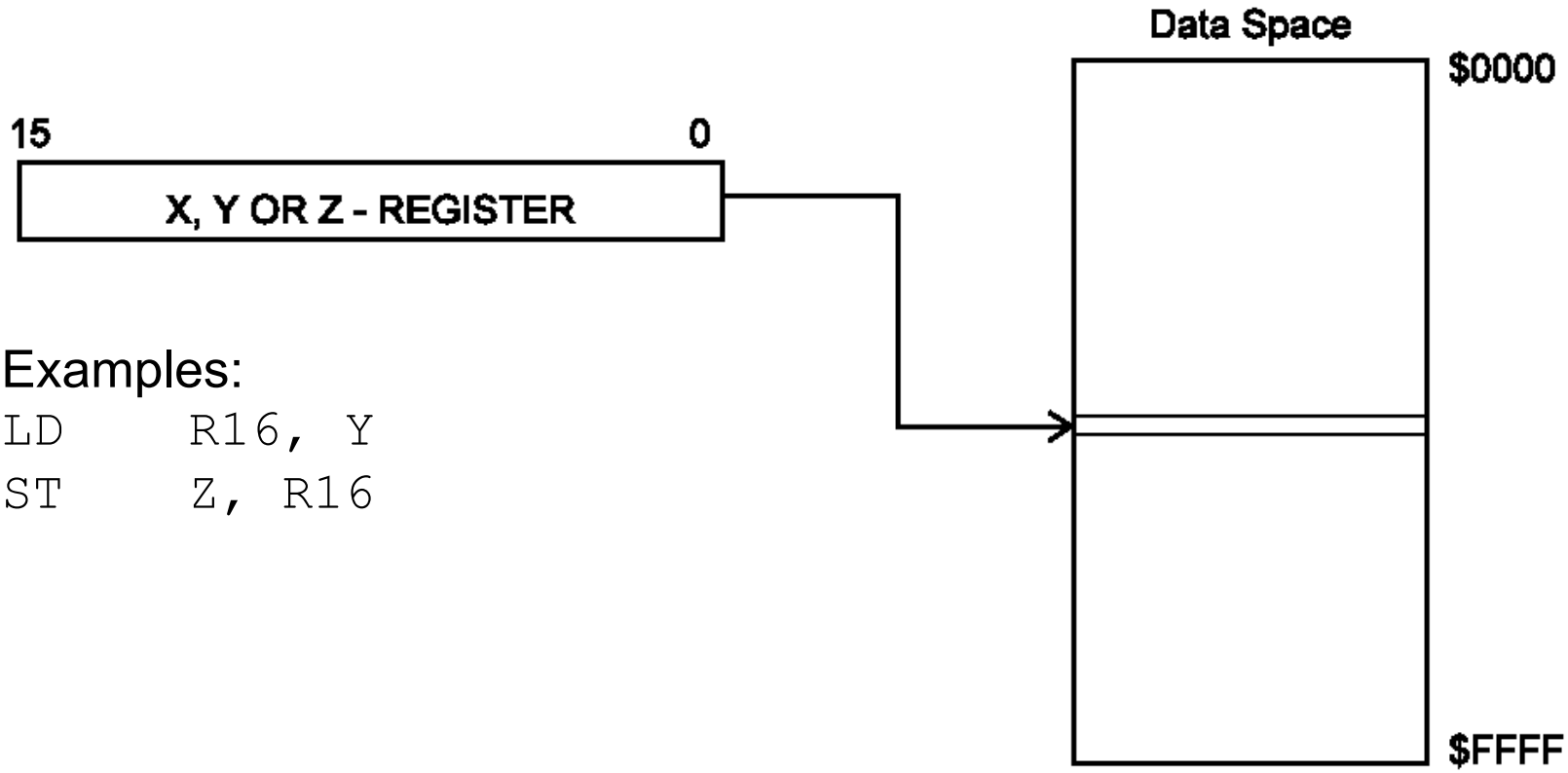
Data Direct



Examples:

```
STS    0x1000,R16
```

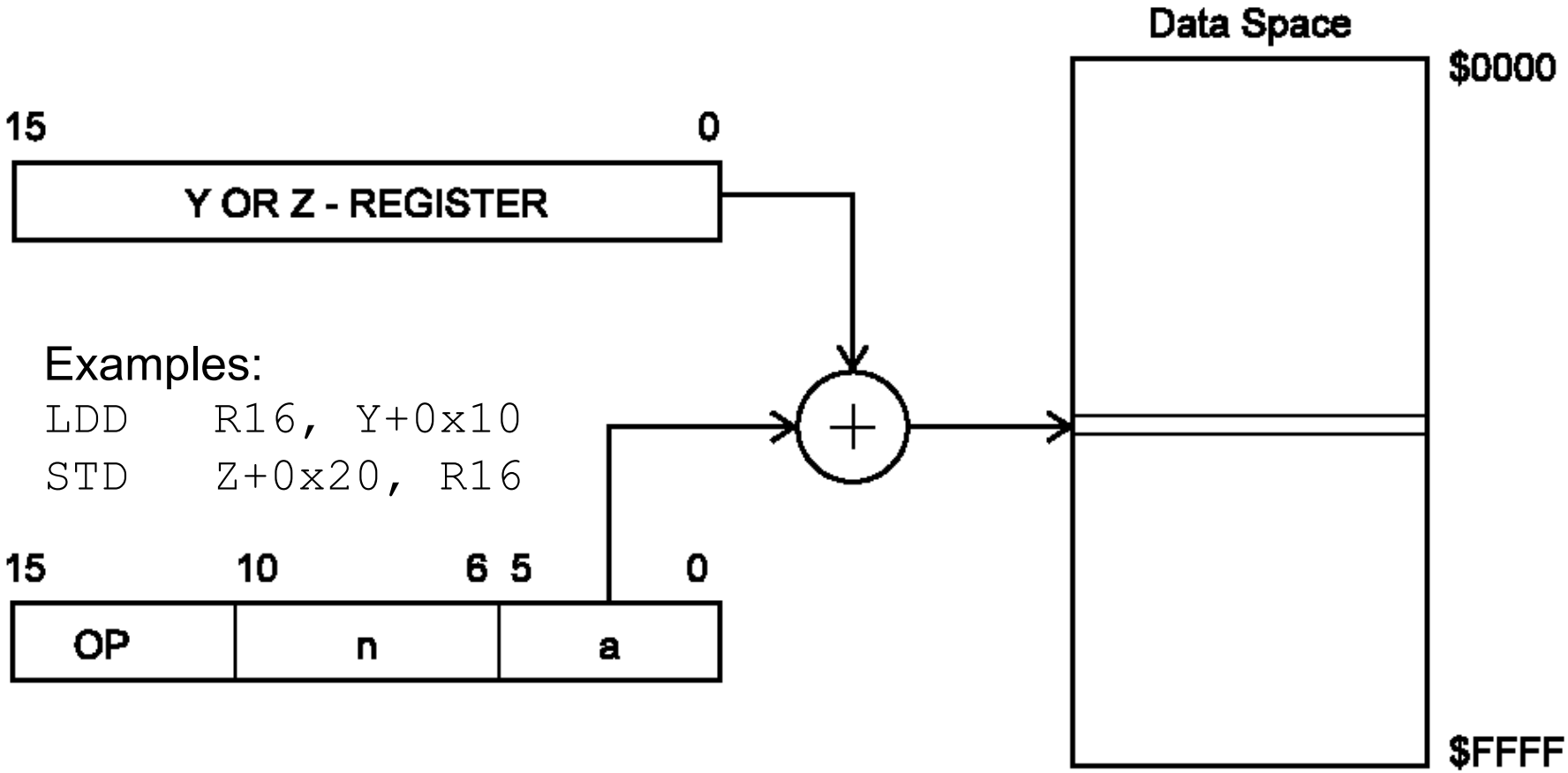
Data Indirect



Examples:

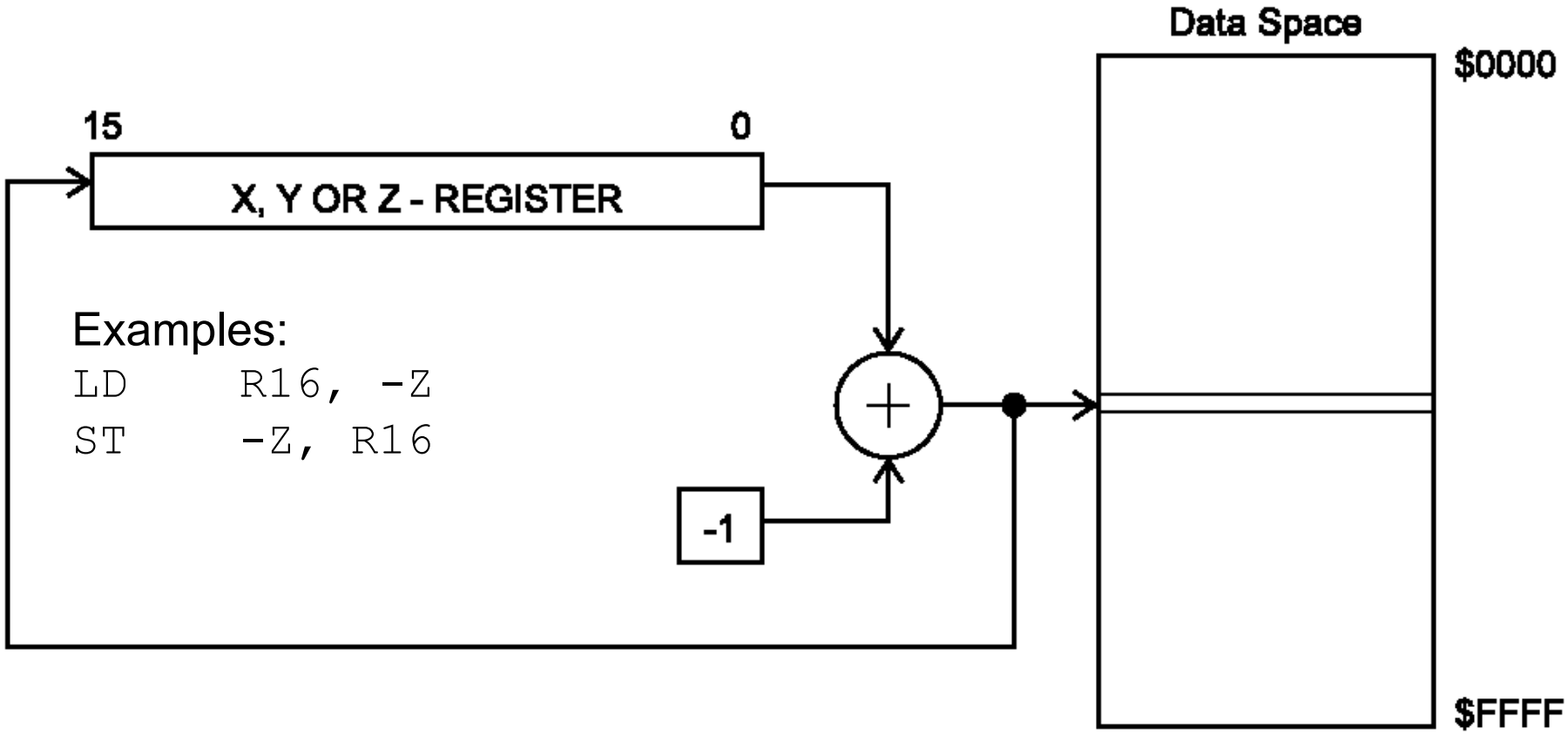
```
LD      R16, Y
ST      Z, R16
```

Data Indirect w/ Displacement



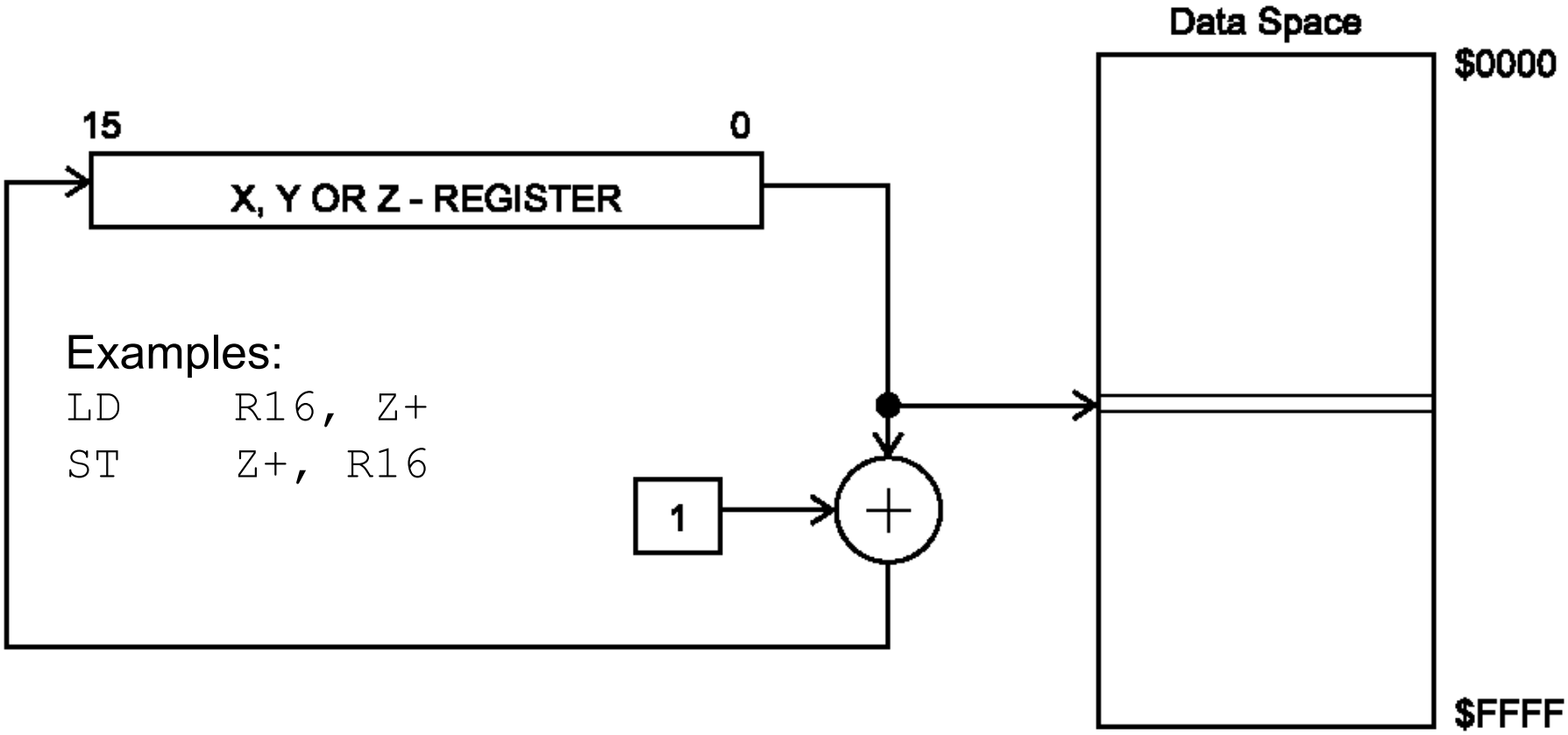
Data Indirect: Pre-Decrement

Embedded System
Design Laboratory



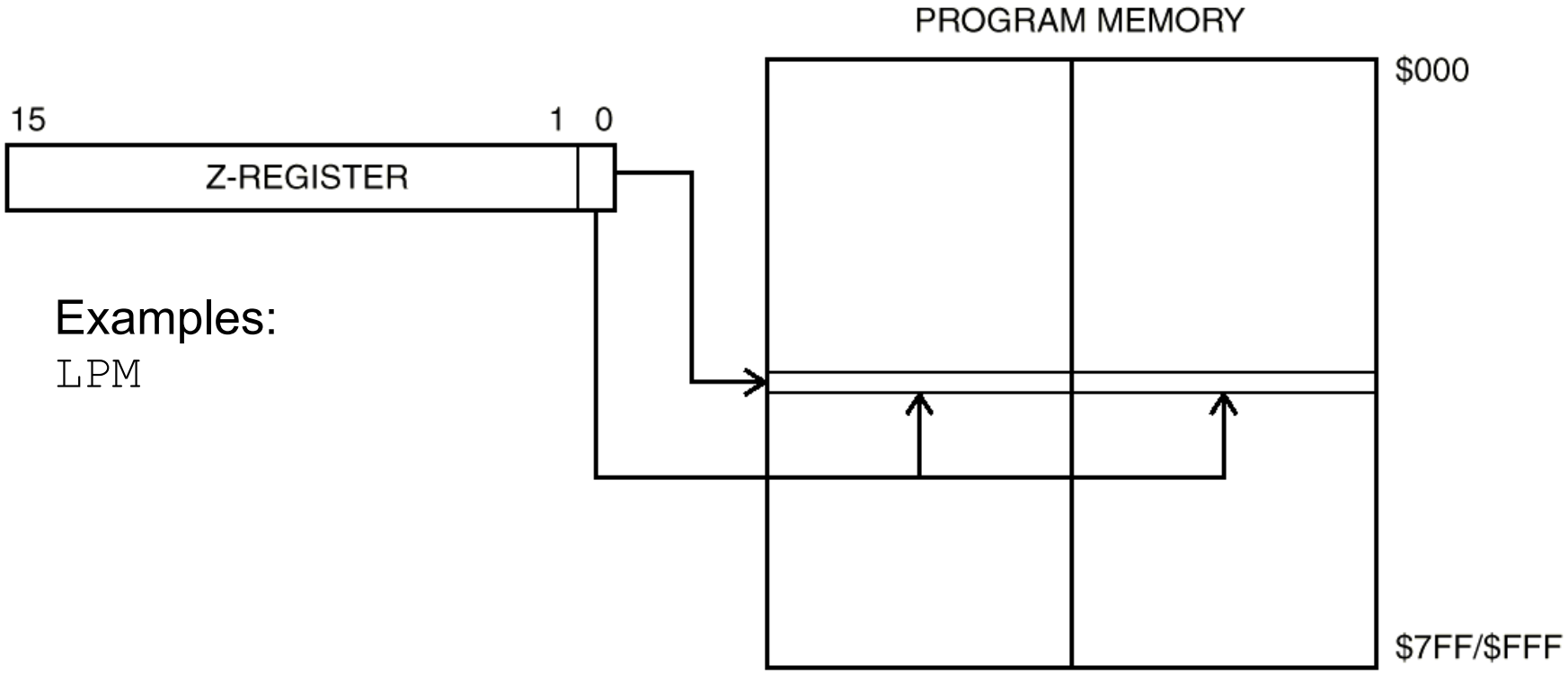
EE281 Data Indirect: Post-Increment

Embedded System
Design Laboratory



Prog. Memory Addressing

Embedded System
Design Laboratory

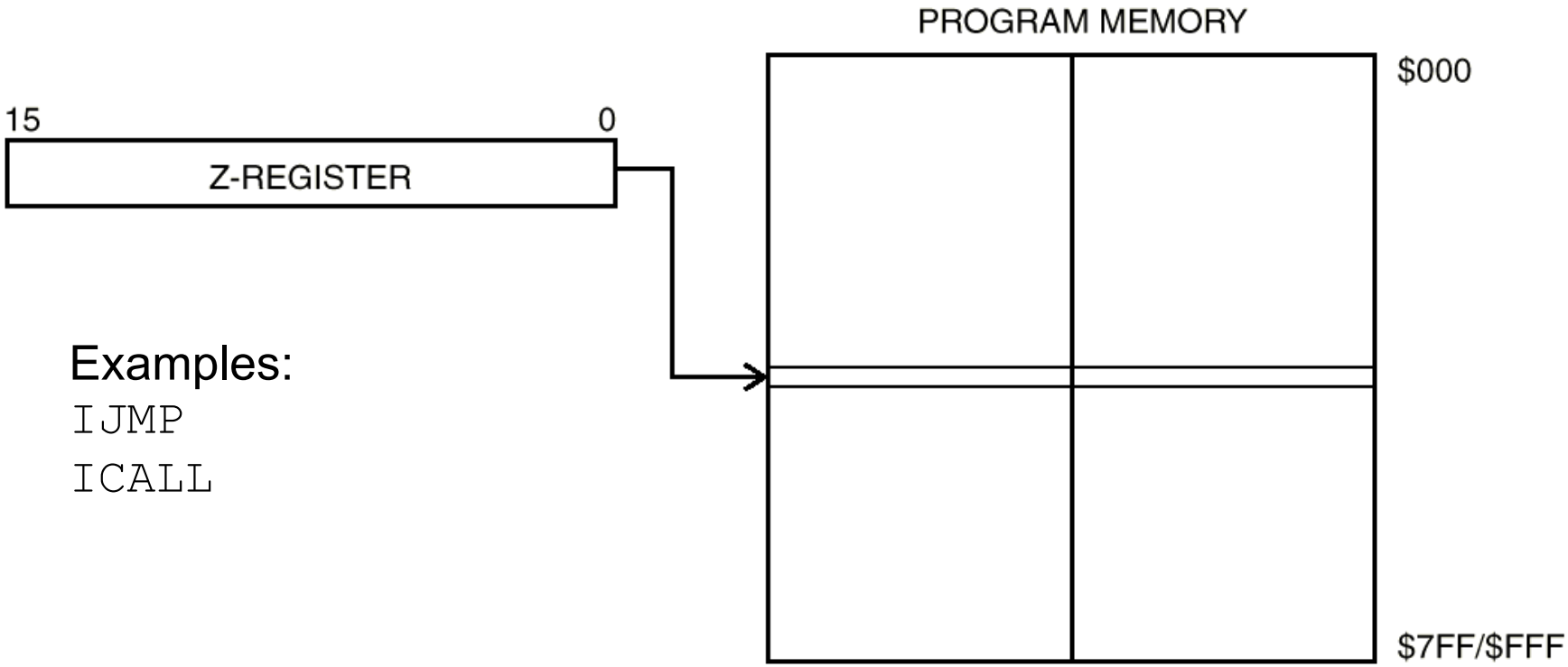


Examples:

`LPM`

Indirect Program Addressing

Embedded System
Design Laboratory

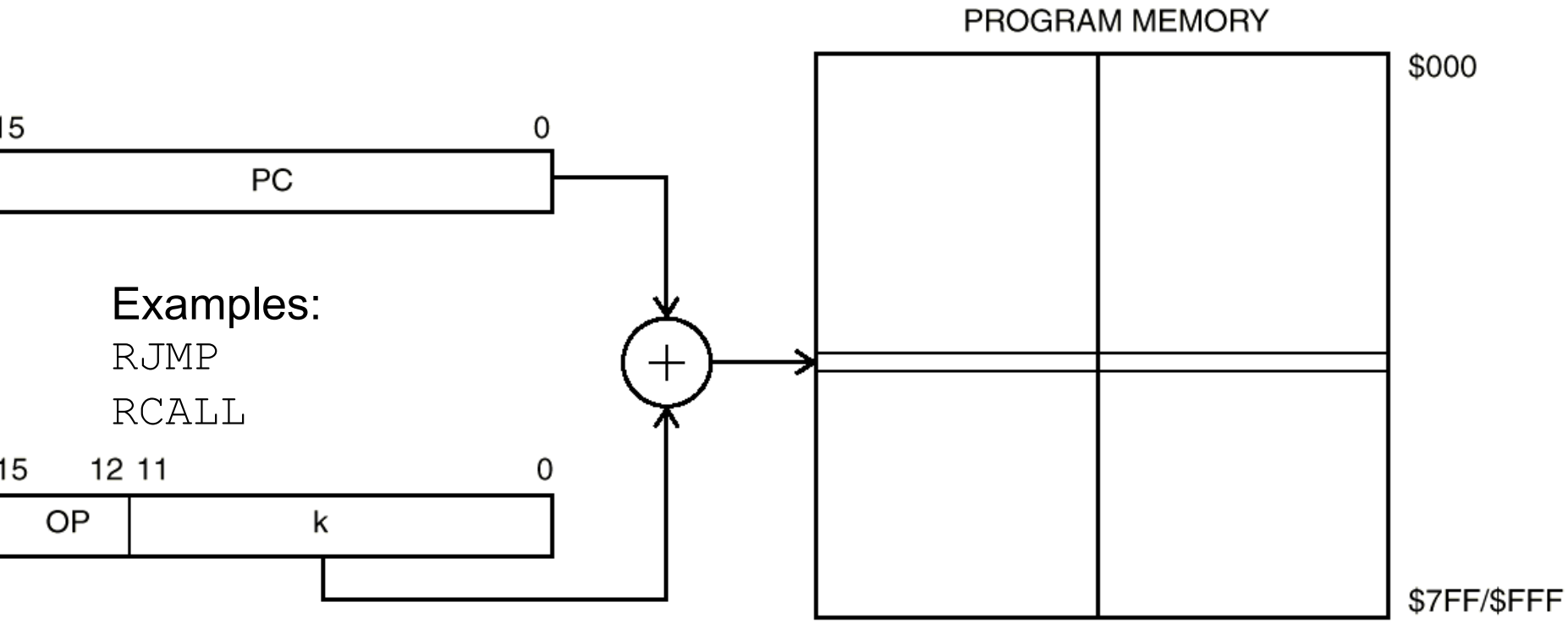


Examples:

```
IJMP
ICALL
```

EE281 Relative Program Addressing

Embedded System
Design Laboratory



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 Rdl, K` – Add Immediate to Word
      - Rdh:Rdl <- Rdh:Rdl + K

EE281

Other Math and Logic

Embedded System
Design Laboratory

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

EE281

Embedded System
Design Laboratory

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/clr
- `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

EE281

Bit and Bit Test Instructions

Embedded System
Design Laboratory

- `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

EE281

Embedded System
Design Laboratory

Other Instructions

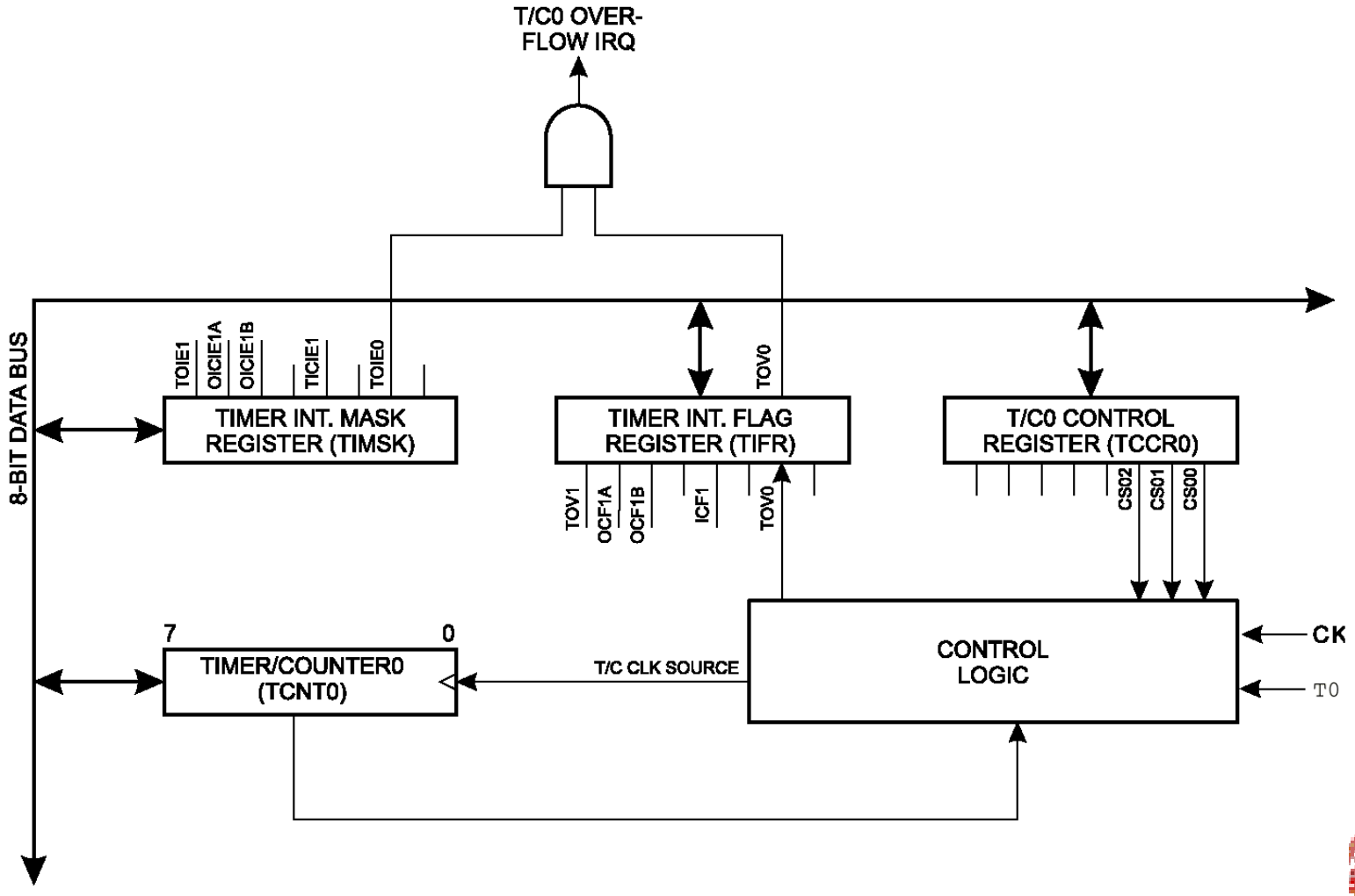
- `NOP` – Do nothing for 1 cycle
- `SLEEP` – Sleep until reset or interrupted
- `WDR` – Watch Dog Reset

AVR Timer/Counter 0

- 8 Bit
- Wrap-Around Up Counter
- Interrupt on Overflow

EE281 AVR Timer/Counter 0 (cont'd)

Embedded System
Design Laboratory

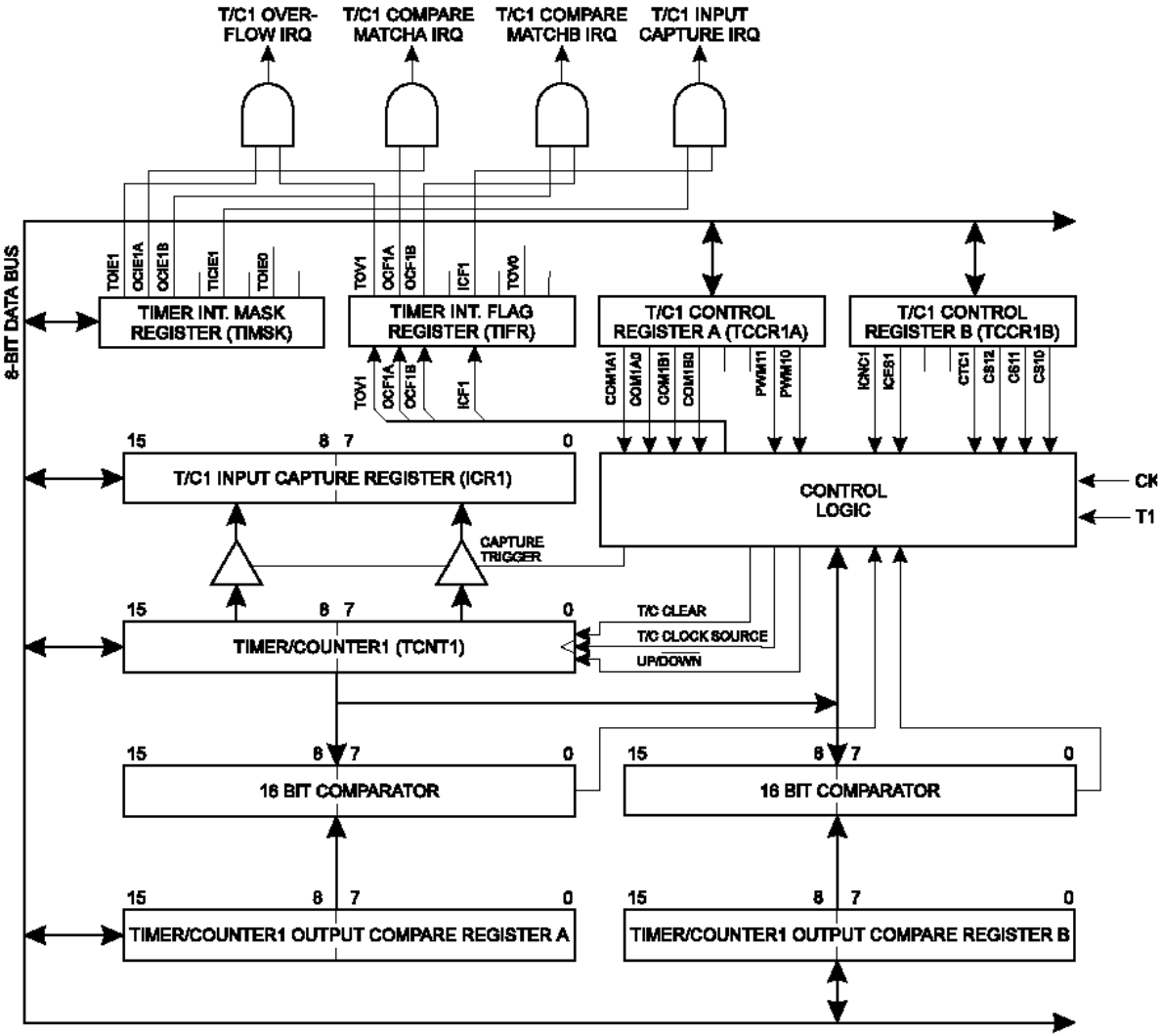


AVR Timer/Counter 1

- 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.

EE281 AVR Timer/Counter 1 (cont'd)

Embedded System Design Laboratory



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).

CS02	CS01	CS00	Description
0	0	0	Stop, the Timer/Counter0 is stopped.
0	0	1	CK
0	1	0	CK/8
0	1	1	CK/64
1	0	0	CK/256
1	0	1	CK/1024
1	1	0	External Pin T0, falling edge
1	1	1	External Pin T0, rising edge

EE281

Embedded System
Design Laboratory

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