Interfacing Sensors and Actuators

How to Interface to Sensors and Actuators

- **Example, adding a sensor to the iRobot**
  - **Starting with a conceptual intention**
  - **Finding the right pin**
  - **ADC & I/O pin electrical properties**
    - **What can drive what, supply V & mA to sensors, motors, audio, LEDs. What is open collector, TTL level.**
  - **Sensor's electrical properties**
    - **Amplifier, optoisolator e.g. 110 VAC or sensitive/HV input**
  - **Mechanical fasteners & electrical connectors**
- Actuators
  - E & M fields
  - Motors: DC, AC, [poly-phase]
  - Drivers: what signals in & out, why PWM, Linear, [Class-D]
  - Robot mechanism, Inverse Kinematics relevance, in EE125

Interfacing an acceleration sensor to the iRobot Create

Dual-Axis Accelerometer Evaluation Board

ADXL322EB

CIRCUIT AND LAYOUT DIAGRAMS

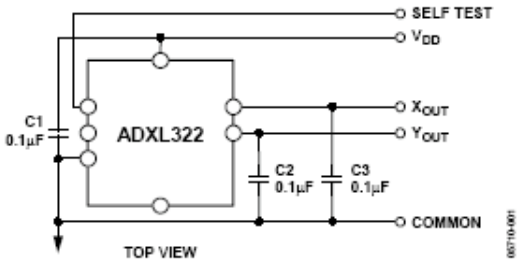


Figure 1. Schematic

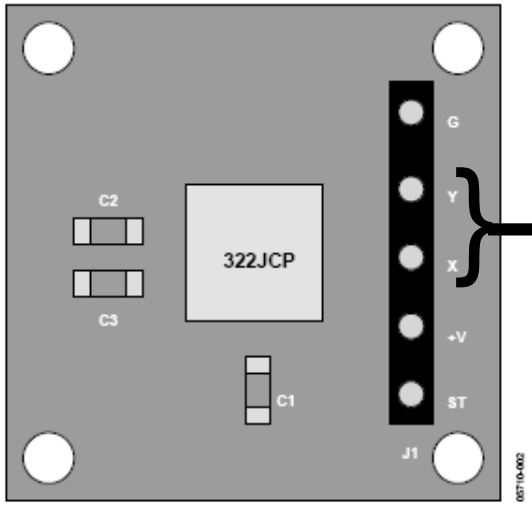
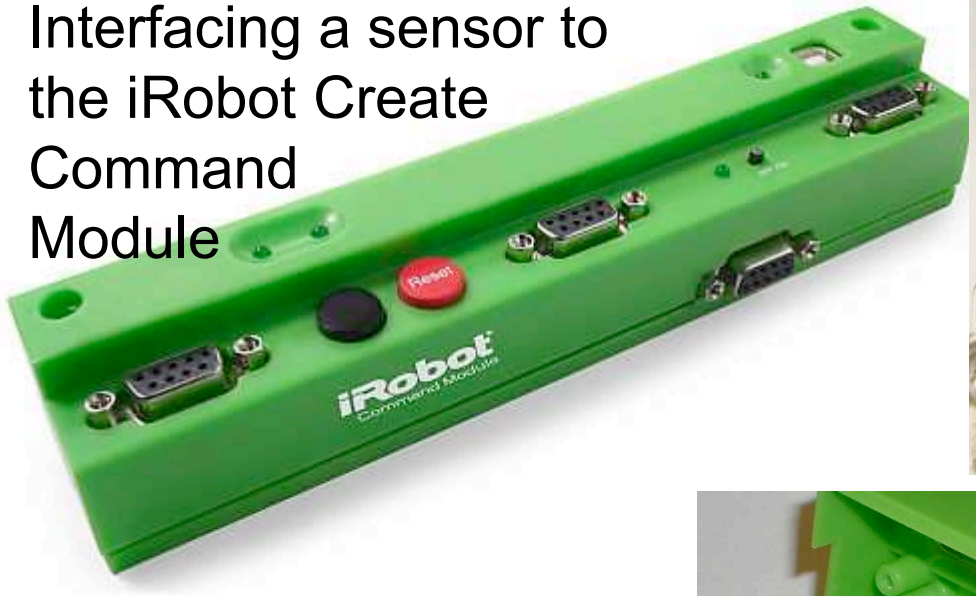
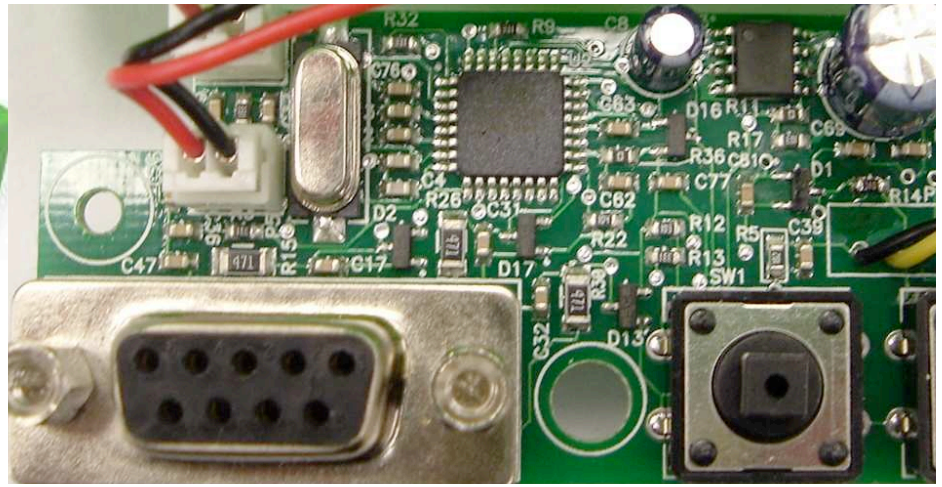


Figure 2. Physical Layout

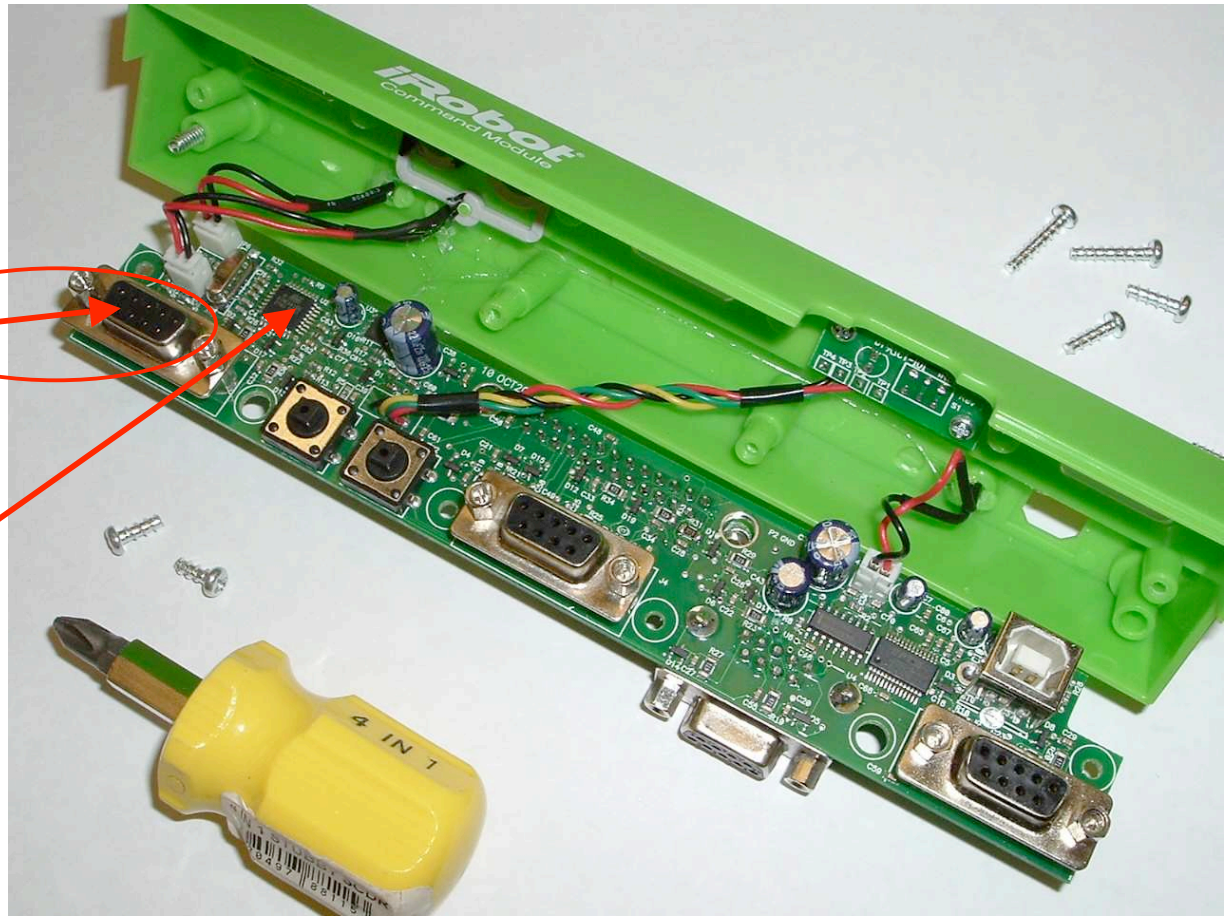
Interfacing a sensor to the iRobot Create Command Module





Command Module
9-pin I/O port
“Top Left ePort”

ATMega 168
Microcontroller



http://www.irobot.com/filelibrary/create/CommandModuleGettingStarted.pdf

Introduction

With the iRobot Command Module, you can control iRobot Create with programs you write in C or C++. And when you add custom hardware to the Command Module and iRobot Create, you can invent almost any kind of robot you want.

The Command Module plugs into iRobot Create's Cargo Bay connector and screws down for a secure connection. Its four DB-9 expansion ports (ePorts) make adding your own sensors, lights, and motors inexpensive and easy. For more detail, download the Command Module Owner's Manual at www.irobot.com/create.

With the programs you write, you can control iRobot Create's motors, lights, and songs and read its sensors using the robot's Open Interface serial protocol (details are in the iRobot Create Owner's Manual and online at www.irobot.com/create). You can also control and read your own custom sensors, buttons, LEDs, and motors when you connect them to the Command Module's expansion ports. Once you write your programs, they are easy to download to the Command Module's microcontroller.

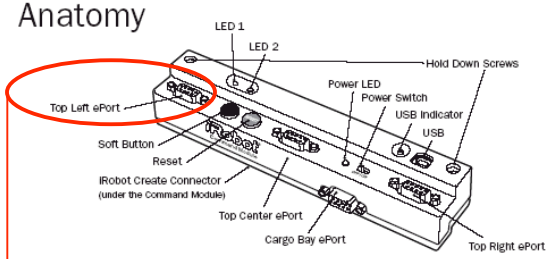
Start with one of the example programs; expand and change it as you wish. For more information on writing software for the Command Module see the Software Reference chapter in the Command Module Owner's Manual and the Open Interface reference guide.

What's included:
iRobot Command Module
USB Cable
iRobot Command Module CD that includes:

- WinAVR install
- USB Driver install
- Example programs
- Atmel AVR ATmega168 microcontroller data sheet

For more details on your iRobot Command Module and to download the Owner's Manual, visit **www.irobot.com/create.**

Anatomy



The tables below show the signal connections for the each of the ePorts. The four ePorts have similar electrical connections, allowing add-on modules to be installed in different ePorts depending upon your needs.

Top Left ePort

Pin#	Description	Name
1	Analog Input 7	ADC7
2	Digital I/O (Port C pin 3) or Analog Input 4	PC3/ADC4
3	Digital I/O (Port B pin 3)	PB3
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Not Connected	NC
9	Low Side Driver 0	LD0

Top Right ePort

Pin#	Description	Name
1	Analog Input 6	ADC6
2	Digital I/O (Port C pin 2) or Analog Input 2	PC2/ADC2
3	Digital I/O (Port B pin 2)	PB2
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Not Connected	NC
9	Low Side Driver 0	LD0

Top Center ePort

Pin#	Description	Name
1	Digital I/O (Port C pin 5) or Analog Input 5	PC5/ADC5
2	Digital I/O (Port C pin 1) or Analog Input 1	PC1/ADC1
3	Digital I/O (Port B pin 1)	PB1
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Not Connected	NC
9	Low Side Driver 0	LD0

Cargo Bay ePort

Pin#	Description	Name
1	Digital I/O (Port C pin 4) or Analog Input 4	PC4/ADC4
2	Digital I/O (Port C pin 0) or Analog Input 0	PC0/ADC0
3	Digital I/O (Port B pin 0)	PB0
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Low Side Driver 1	LD1
9	Low Side Driver 2	LD2

Processor I/O Pins

The Command Module's Atmel AVR ATmega 168 microcontroller has 23 I/O pins, arranged into 3 ports (B, C and D), plus 2 additional analog input pins. The table below gives a description of each pin along with the location of the hardware to which it is connected. For information on how to control and read from these pins, see the Command Module Owner's Manual.

Pin Name	Description	Location
PB0	Digital I/O	Cargo Bay ePort pin 3
PB1	Digital I/O	Top Center ePort pin 3
PB2	Digital I/O	Top Right ePort pin 2
PB3	Digital I/O	Top Left ePort pin 3
PB4	Serial port connector select. 1 = USB port. 0 = iRobot Create	Internal
PB5	iRobot Create Power Detect. High if iRobot Create is on.	iRobot Create connector pins 10-13
PB6	Clock line	Internal Use only
PB7	Clock line	Internal Use only
PC0	Digital I/O or Analog Input	Cargo Bay ePort pin 2
PC1	Digital I/O or Analog Input	Top Center ePort pin 2
PC2	Digital I/O or Analog Input	Top Right ePort pin 2
PC3	Digital I/O or Analog Input	Top Left ePort pin 2
PC4	Digital I/O or Analog Input	Cargo Bay ePort pin 1
PC5	Digital I/O or Analog Input	Top Center ePort pin 2
PC6	Reset Line	Internal Use only
PD0	Serial Rx	iRobot Create connector pin 2 or USB
PD1	Serial Tx	iRobot Create connector pin 1 or USB
PD2	iRobot Create Device Detect Input	iRobot Create connector pin 15
PD3	USB Detect	USB port
PD4	Command Module Soft Button	Left button
PD5	Command Module LED 1	Left green LED
PD6	Command Module LED 2	Right green LED
PD7	iRobot Create Power Toggle (on rising edge)	iRobot Create connector pin 3
ADC6	Analog Input	Top Right ePort pin 1
ADC7	Analog Input	Top Left ePort pin 1

System Requirements:
Windows XP*
USB connection

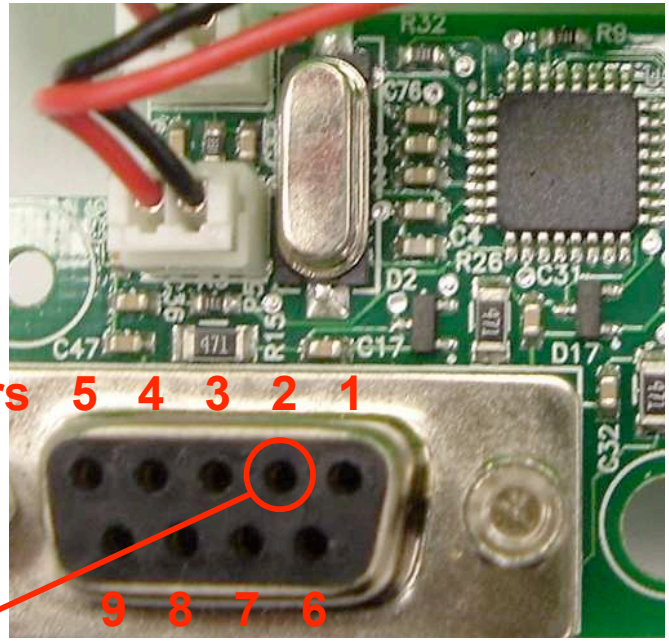
*Support for additional operating systems may be available at **www.irobot.com/create.**

Top Left ePort

Pin#	Description	Name
1	Analog Input 7	ADC7
2	Digital I/O (Port C pin 3) or Analog Input 4	PC3/ADC4
3	Digital I/O (Port B pin 3)	PB3
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Not Connected	NC
9	Low Side Driver 0	LD0

Cargo Bay ePort

Pin#	Description	Name
1	Digital I/O (Port C pin 4) or Analog Input 4	PC4/ADC4
2	Digital I/O (Port C pin 0) or Analog Input 0	PC0/ADC0
3	Digital I/O (Port B pin 0)	PB0
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Low Side Driver 1	LD1
9	Low Side Driver 2	LD2



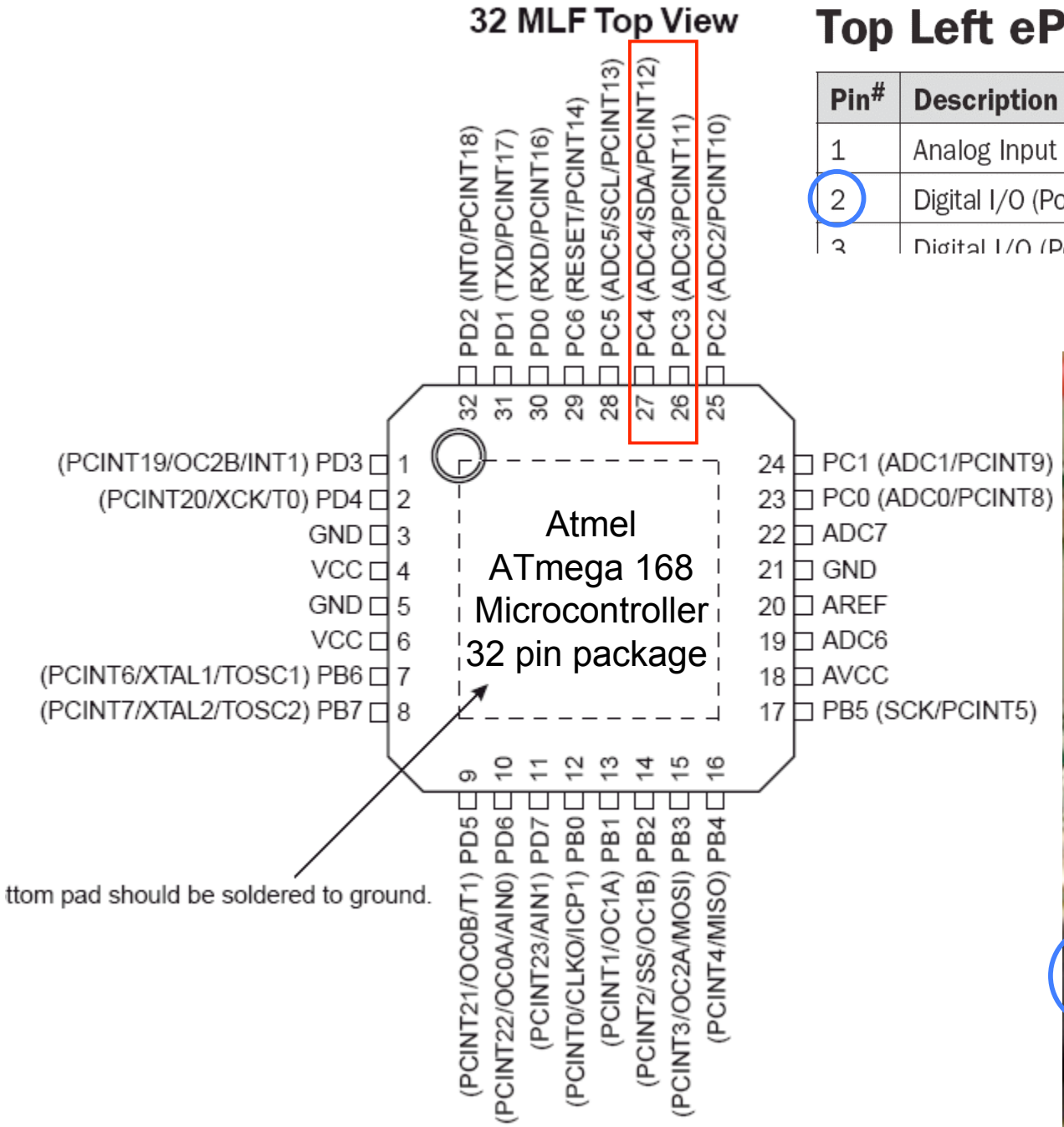
- ATMega 168

Top Left ePort

Pin#	Description	Name
1	Analog Input 7	ADC7
2	Digital I/O (Port C pin 3) or Analog Input 4	PC3/ADC4
3	Digital I/O (Port B pin 3)	PB3
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Not Connected	NC
9	Low Side Driver 0	LD0

Cargo Bay ePort

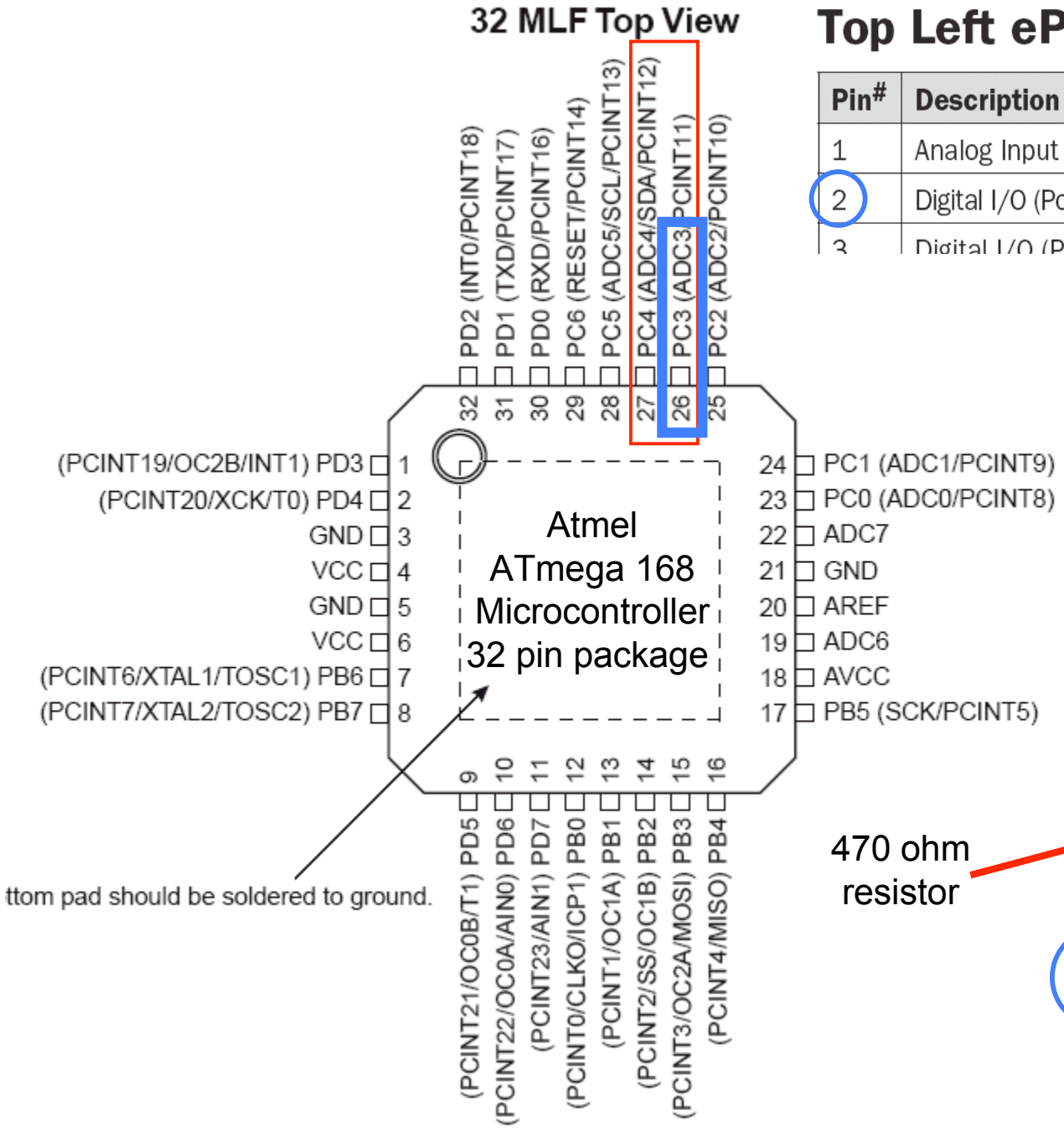
Pin#	Description	Name
1	Digital I/O (Port C pin 4) or Analog Input 4	PC4/ADC4
2	Digital I/O (Port C pin 0) or Analog Input 0	PC0/ADC0
3	Digital I/O (Port B pin 0)	PB0
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Low Side Driver 1	LD1
9	Low Side Driver 2	LD2



Top Left ePort

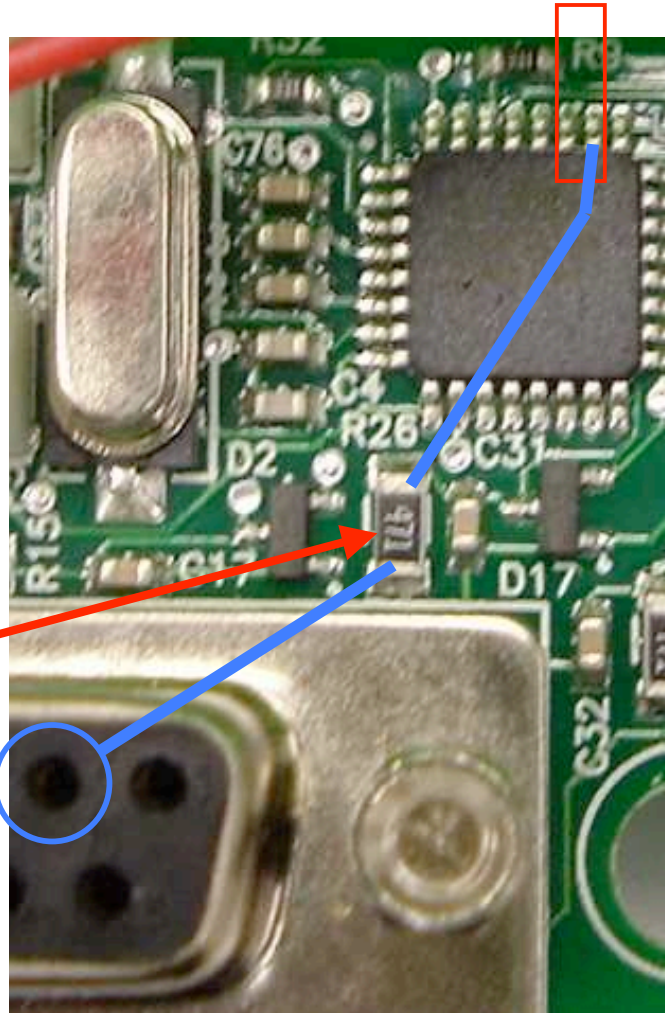
Pin#	Description	Name
1	Analog Input 7	ADC7
2	Digital I/O (Port C pin 3) or Analog Input 4	PC3/ADC4
3	Digital I/O (Port B pin 3)	PB3

http://www.atmel.com/dyn/resources/prod_documents/doc2545.pdf



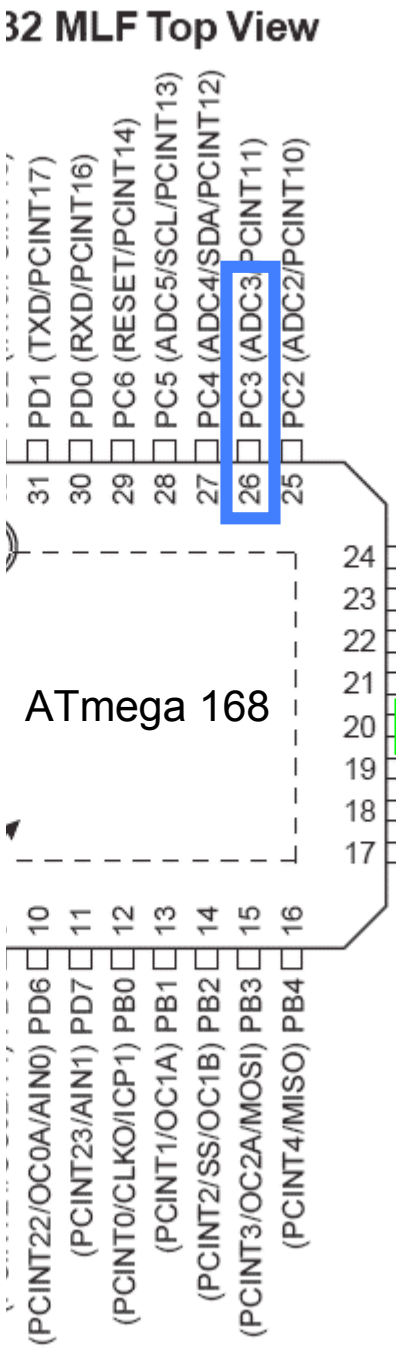
Top Left ePort

Pin#	Description	Name
1	Analog Input 7	ADC7
2	Digital I/O (Port C pin 3) or ~~Analog Input 4~~ (Analog Input 3)	PC3/~~ADC4~~ (ADC3)
3	Digital I/O (Port B pin 3)	PB3



http://www.atmel.com/dyn/resources/prod_documents/doc2545.pdf

http://www.atmel.com/dyn/resources/prod_documents/doc2545.pdf page 312



28.8 ADC Characteristics

Table 28-7. ADC Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
	Resolution			10		Bits
	Absolute accuracy (Including INL, DNL, quantization error, gain and offset error)	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		2		LSB
		V_{REF} = 4V, V_{CC} = 4V, ADC clock = 1 MHz		4.5		LSB
		V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz Noise Reduction Mode		2		LSB
		V_{REF} = 4V, V_{CC} = 4V, ADC clock = 1 MHz Noise Reduction Mode		4.5		LSB
	Integral Non-Linearity (INL)	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		0.5		LSB
	Differential Non-Linearity (DNL)	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		0.25		LSB
	Gain Error	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		2		LSB
	Offset Error	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		2		LSB
	Conversion Time	Free Running Conversion	13		260	µs
	Clock Frequency		50		1000	kHz
AV_{CC} (1)	Analog Supply Voltage		V_{CC} - 0.3		V_{CC} + 0.3	V
V_{REF}	Reference Voltage		1.0		AV_{CC}	V
V_{IN}	Input Voltage		GND		V_{REF}	V
	Input Bandwidth			38.5		kHz
V_{INT}	Internal Voltage Reference		1.0	1.1	1.2	V
R_{REF}	Reference Input Resistance			32		kΩ
R_{AIN}	Analog Input Resistance			100		MΩ

Note: 1. AV_{CC} absolute min/max: 1.8V/5.5V

SPECIFICATIONS

T_A = 25°C, V_S = 3 V, C_X = C_Y = 0.1 µF, Acceleration = 0 *g*, unless otherwise noted[1].

Table 1.

Parameter	Conditions	Min	Typ	Max	Unit
SENSOR INPUT	Each axis				
Measurement Range			±2		*g*
Nonlinearity	% of full scale		±0.2		%
Package Alignment Error			±1		Degrees
Alignment Error	X sensor to Y sensor		±0.1		Degrees
Cross-Axis Sensitivity			±2		%
SENSITIVITY (RATIOMETRIC)[2]	Each axis				
Sensitivity at X_{OUT}, Y_{OUT}	V_S = 3 V	378	420	462	mV/*g*
Sensitivity Change due to Temperature[3]	V_S = 3 V		0.01		%/°C
ZERO *g* BIAS LEVEL (RATIOMETRIC)	Each axis				
0 *g* Voltage at X_{OUT}, Y_{OUT}	V_S = 3 V	1.3	1.5	1.7	V
Initial 0 *g* Bias Deviation from Ideal			±50		mg
0 *g* Offset Vs. Temperature			<±0.5		mg/°C
NOISE PERFORMANCE					
Noise Density	at 25°C		220		µg/√Hz rms
FREQUENCY RESPONSE[4]					
C_X, C_Y Range[5]		0.002		10	µF
R_{FILT} Tolerance			32 ± 15%		kΩ
Sensor Resonant Frequency			5.5		kHz
SELF-TEST[6]					
Logic Input Low			0.6		V
Logic Input High			2.4		V
ST Input Resistance to Ground			50		kΩ
Output Change at X_{OUT}, Y_{OUT}	Self-test 0 to 1		125		mV
OUTPUT AMPLIFIER					
Output Swing Low	No load		0.2		V
Output Swing High	No load		2.7		V
POWER SUPPLY					
Operating Voltage Range		2.4		6	V
Quiescent Supply Current			0.45		mA
Turn-On Time[7]			20		ms
TEMPERATURE					
Operating Temperature Range		–20		70	°C

[1] All minimum and maximum specifications are guaranteed. Typical specifications are not guaranteed.
[2] Sensitivity is essentially ratiometric to V_S. For V_S = 2.7 V to 3.3 V, sensitivity is 138 mV/V/*g* to 142 mV/V/*g* typical.
[3] Defined as the output change from ambient-to-maximum temperature or ambient-to-minimum temperature.
[4] Actual frequency response controlled by user-supplied external capacitor (C_X, C_Y).
[5] Bandwidth = 1/(2 × π × 32 kΩ × C). For C_X, C_Y = 0.002 µF, bandwidth = 2500 Hz. For C_X, C_Y = 10 µF, bandwidth = 0.5 Hz. Minimum/maximum values are not tested.
[6] Self-test response changes cubically with V_S.
[7] Larger values of C_X, C_Y increase turn-on time. Turn-on time is approximately 160 × C_X or C_Y + 4 ms, where C_X, C_Y are in µF.

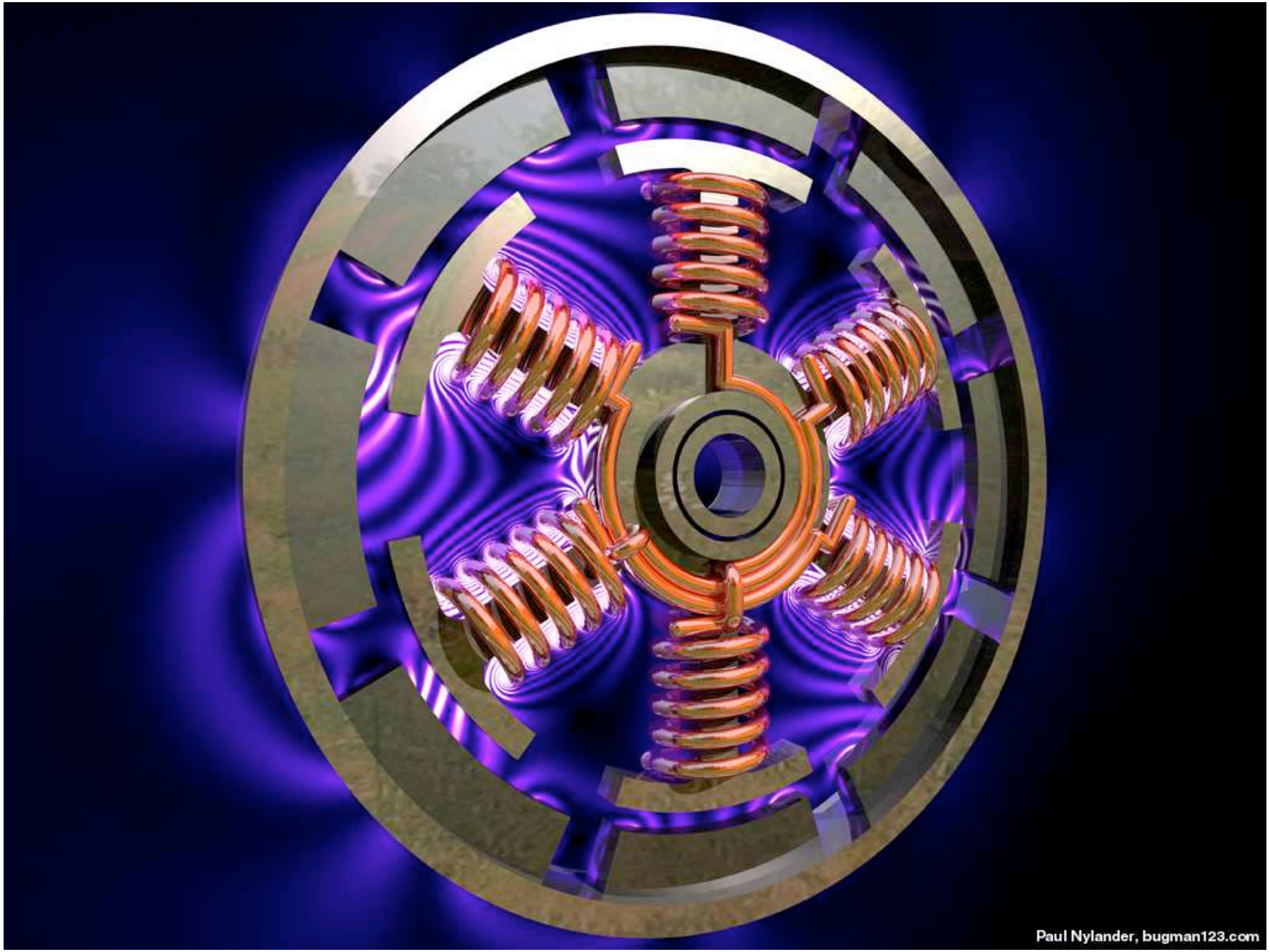
How to Interface to Sensors and Actuators

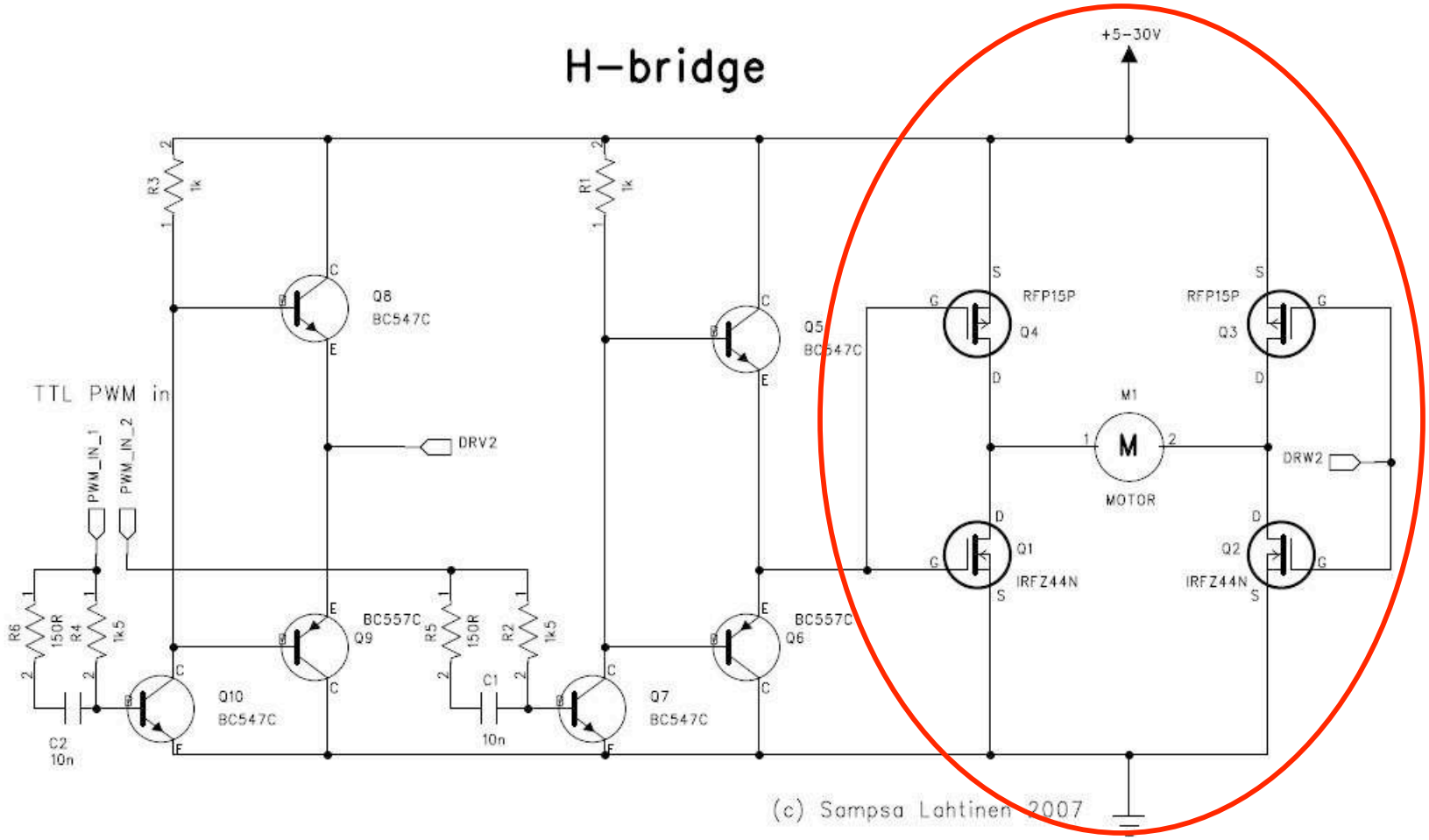
- Example, adding a sensor to the iRobot
  - Starting with a conceptual intention
  - Finding the right pin
  - ADC & I/O pin electrical properties
    - What can drive what, supply V & mA to sensors, motors, audio, LEDs. What is open collector, TTL level.
  - Sensor's electrical properties
    - Amplifier, optoisolator e.g. 110 VAC or sensitive/HV input
  - Mechanical fasteners & electrical connectors
- **Actuators**
  - **E & M fields**
  - **Motors: DC, AC, [poly-phase]**
  - **Drivers: what signals in & out, why PWM, Linear, [Class-D]**
  - **Robot mechanism, Inverse Kinematics relevance, in EE125**

Motors

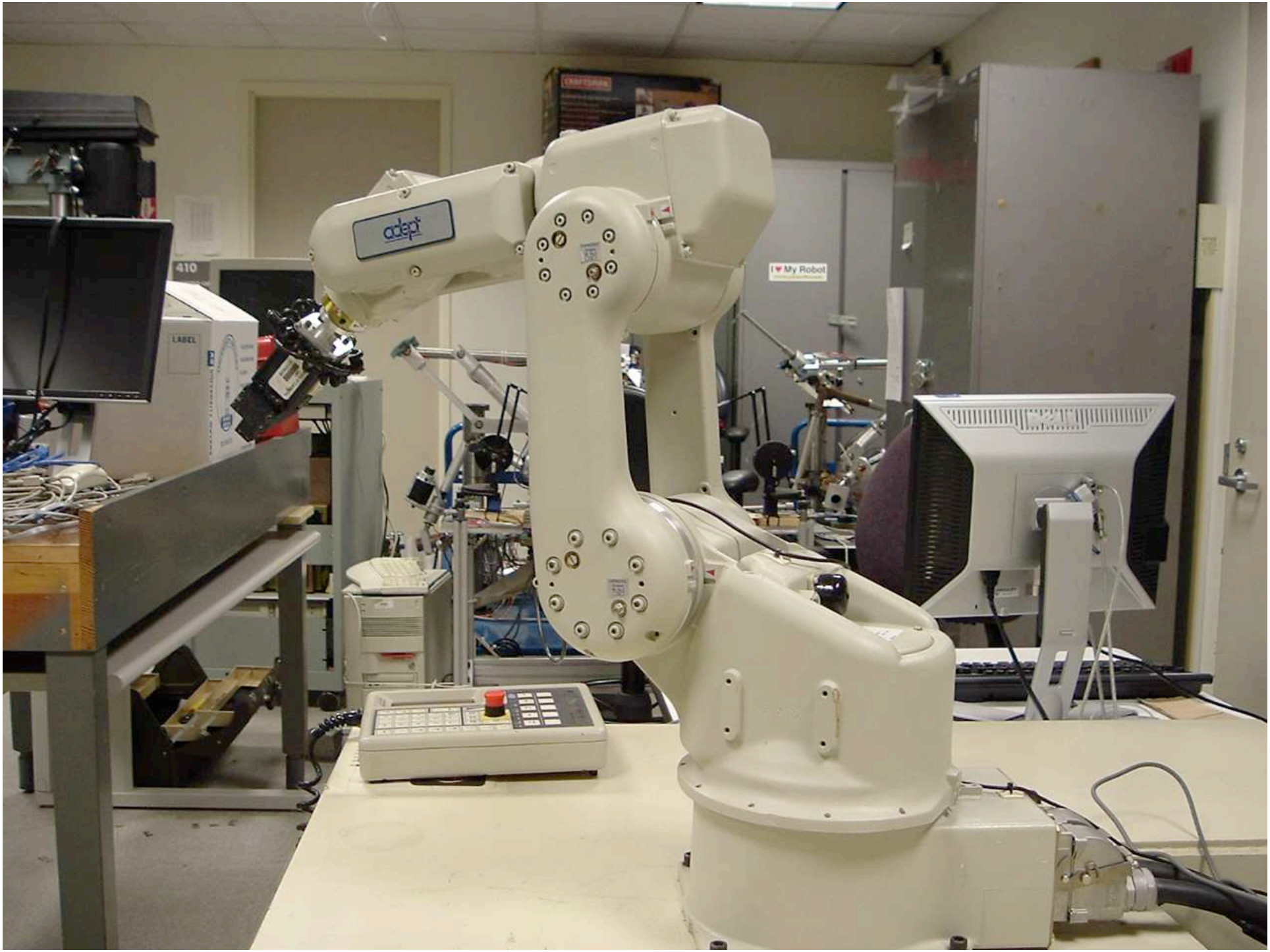
http://en.wikipedia.org/wiki/Electric_motor

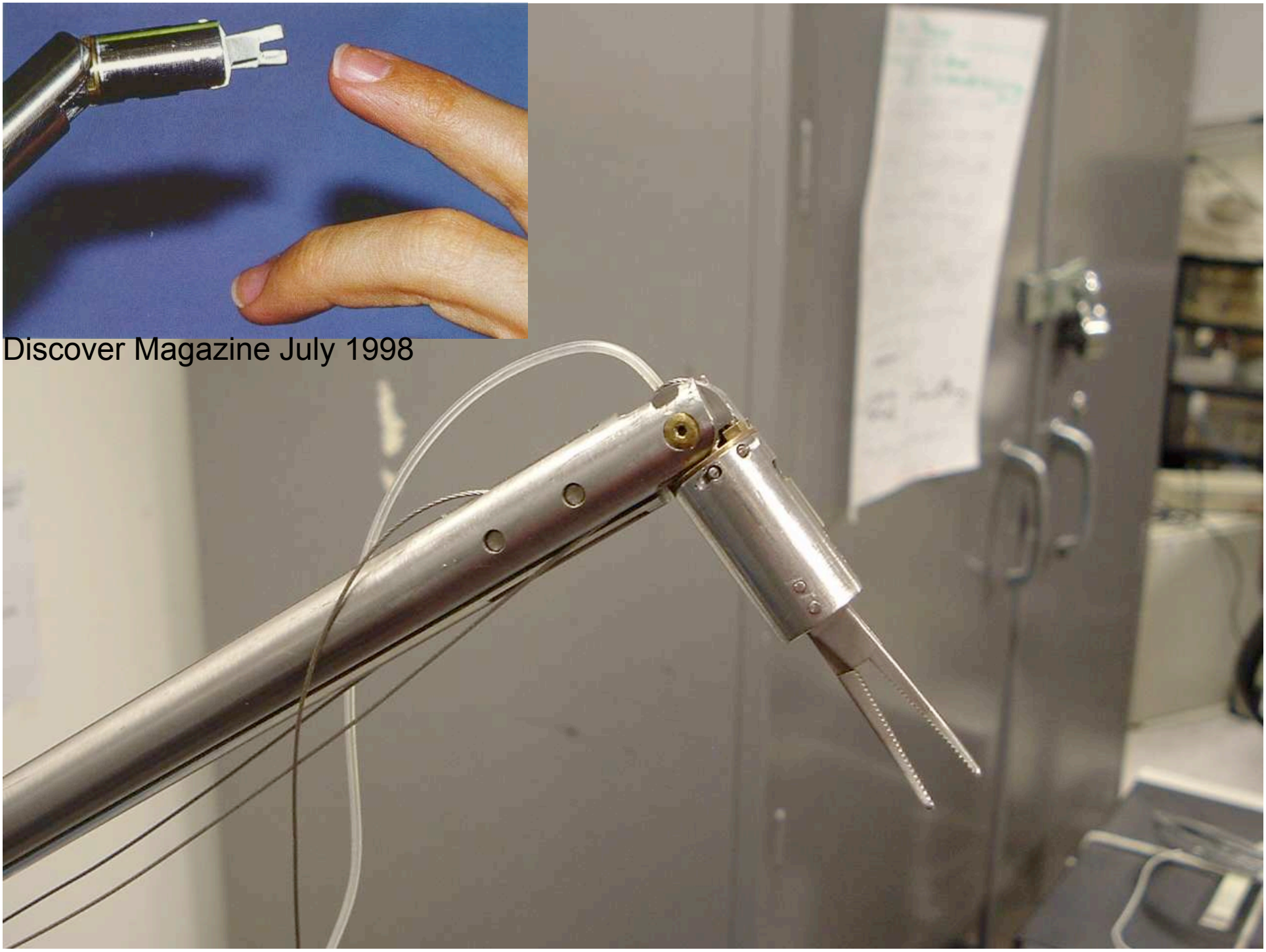
Magnetic Fields & Forces



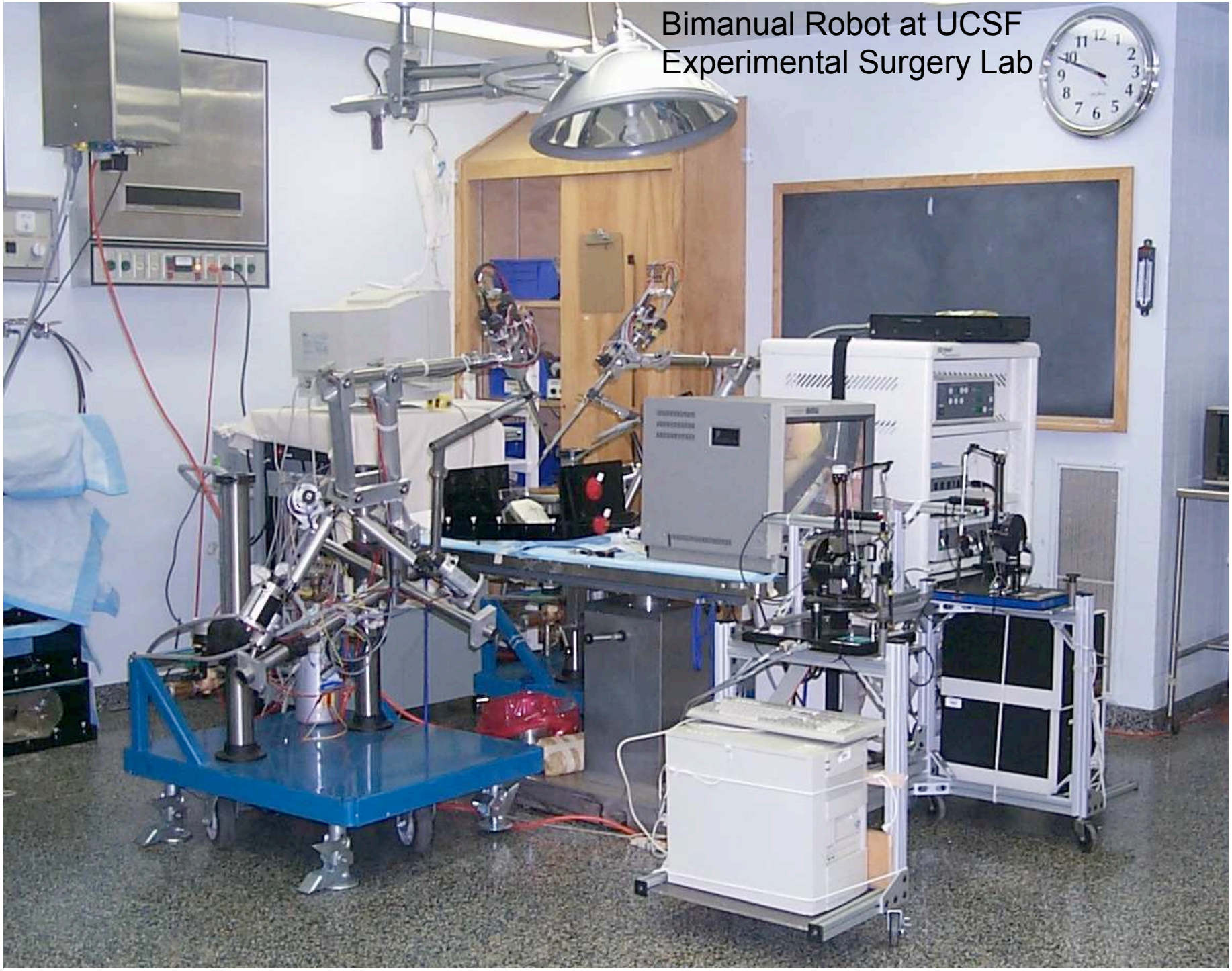


http://www.kolumbus.fi/st_hm.lahtinen/Elektroniikka/H-bridge.jpg


adept
410
LABEL
I ♥ My Robot



Discover Magazine July 1998


Bimanual Robot at UCSF
Experimental Surgery Lab

Extra slides below…

Dual-Axis Accelerometer Evaluation Board

ADXL322EB

CIRCUIT AND LAYOUT DIAGRAMS

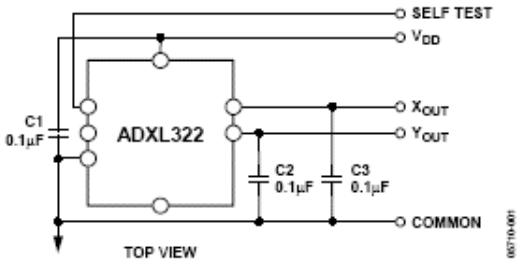


Figure 1. Schematic

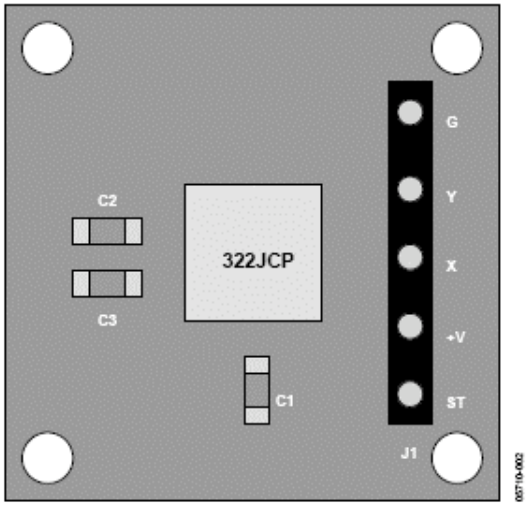


Figure 2. Physical Layout

28. Electrical Characteristics

28.1 Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0 mA
DC Current V_{CC} and GND Pins	200.0 mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

28.2 DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
V_{IL}	Input Low Voltage, except XTAL1 and $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	-0.5 -0.5		$0.2V_{CC}$[1] $0.3V_{CC}$[1]	V
V_{IH}	Input High Voltage, except XTAL1 and $\overline{\text{RESET}}$ pins	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.7V_{CC}$[2] $0.6V_{CC}$[2]		V_{CC} + 0.5 V_{CC} + 0.5	V
V_{IL1}	Input Low Voltage, XTAL1 pin	V_{CC} = 1.8V - 5.5V	-0.5		$0.1V_{CC}$[1]	V
V_{IH1}	Input High Voltage, XTAL1 pin	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.8V_{CC}$[2] $0.7V_{CC}$[2]		V_{CC} + 0.5 V_{CC} + 0.5	V
V_{IL2}	Input Low Voltage, $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 5.5V	-0.5		$0.2V_{CC}$[1]	V
V_{IH2}	Input High Voltage, $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 5.5V	$0.9V_{CC}$[2]		V_{CC} + 0.5	V
V_{IL3}	Input Low Voltage, $\overline{\text{RESET}}$ pin as I/O	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	-0.5 -0.5		$0.2V_{CC}$[1] $0.3V_{CC}$[1]	V
V_{IH3}	Input High Voltage, $\overline{\text{RESET}}$ pin as I/O	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.7V_{CC}$[2] $0.6V_{CC}$[2]		V_{CC} + 0.5 V_{CC} + 0.5	V
V_{OL}	Output Low Voltage[3], $\overline{\text{RESET}}$ pin as I/O	I_{OL} = 20mA, V_{CC} = 5V I_{OL} = 6mA, V_{CC} = 3V			0.7 0.5	V
V_{OH}	Output High Voltage[4], $\overline{\text{RESET}}$ pin as I/O	I_{OH} = -20mA, V_{CC} = 5V I_{OH} = -10mA, V_{CC} = 3V	4.2 2.3			V
I_{IL}	Input Leakage Current I/O Pin	V_{CC} = 5.5V, pin low (absolute value)			1	µA
I_{IH}	Input Leakage Current I/O Pin	V_{CC} = 5.5V, pin high (absolute value)			1	µA
R_{RST}	Reset Pull-up Resistor		30		60	kΩ
R_{PU}	I/O Pin Pull-up Resistor		20		50	kΩ

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted) (Continued)

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
I_{CC}	Power Supply Current[5]	Active 1MHz, V_{CC} = 2V (ATmega48/88/168V)			0.55	mA
		Active 4MHz, V_{CC} = 3V (ATmega48/88/168L)			3.5	mA
		Active 8MHz, V_{CC} = 5V (ATmega48/88/168)			12	mA
		Idle 1MHz, V_{CC} = 2V (ATmega48/88/168V)		0.25	0.5	mA
		Idle 4MHz, V_{CC} = 3V (ATmega48/88/168L)			1.5	mA
		Idle 8MHz, V_{CC} = 5V (ATmega48/88/168)			5.5	mA
	Power-down mode	WDT enabled, V_{CC} = 3V		8	15	µA
		WDT disabled, V_{CC} = 3V		1	2	µA
V_{ACIO}	Analog Comparator Input Offset Voltage	V_{CC} = 5V V_{in} = V_{CC}/2		10	40	mV
I_{ACLK}	Analog Comparator Input Leakage Current	V_{CC} = 5V V_{in} = V_{CC}/2	-50		50	nA
t_{ACID}	Analog Comparator Propagation Delay	V_{CC} = 2.7V V_{CC} = 4.0V		750 500		ns

Notes:
1. "Max" means the highest value where the pin is guaranteed to be read as low
2. "Min" means the lowest value where the pin is guaranteed to be read as high
3. Although each I/O port can sink more than the test conditions (20 mA at V_{CC} = 5V, 10 mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   ATmega48/88/168:
   1] The sum of all I_{OL}, for ports C0 - C5, ADC7, ADC6 should not exceed 100 mA.
   2] The sum of all I_{OL}, for ports B0 - B5, D5 - D7, XTAL1, XTAL2 should not exceed 100 mA.
   3] The sum of all I_{OL}, for ports D0 - D4, $\overline{\text{RESET}}$ should not exceed 100 mA.
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.
4. Although each I/O port can source more than the test conditions (20 mA at V_{CC} = 5V, 10 mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   ATmega48/88/168:
   1] The sum of all I_{OH}, for ports C0 - C5, D0- D4, ADC7, $\overline{\text{RESET}}$ should not exceed 150 mA.
   2] The sum of all I_{OH}, for ports B0 - B5, D5 - D7, ADC6, XTAL1, XTAL2 should not exceed 150 mA.
   If I_{OH} exceeds the test condition, V_{OH} may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.
5. Values with "Minimizing Power Consumption" enabled (0xFF).

Introduction

With the iRobot Command Module, you can control iRobot Create with programs you write in C or C++. And when you add custom hardware to the Command Module and iRobot Create, you can invent almost any kind of robot you want.

The Command Module plugs into iRobot Create's Cargo Bay connector and screws down for a secure connection. Its four DB-9 expansion ports (ePorts) make adding your own sensors, lights, and motors inexpensive and easy. For more detail, download the Command Module Owner's Manual at www.irobot.com/create.

With the programs you write, you can control iRobot Create's motors, lights, and songs and read its sensors using the robot's Open Interface serial protocol (details are in the iRobot Create Owner's Manual and online at www.irobot.com/create). You can also control and read your own custom sensors, buttons, LEDs, and motors when you connect them to the Command Module's expansion ports. Once you write your programs, they are easy to download to the Command Module's microcontroller.

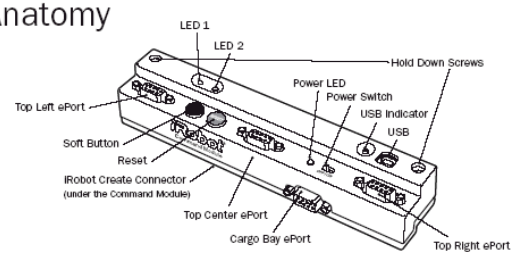
Start with one of the example programs; expand and change it as you wish. For more information on writing software for the Command Module see the Software Reference chapter in the Command Module Owner's Manual and the Open Interface reference guide.

What's included:
iRobot Command Module
USB Cable
iRobot Command Module CD that includes:

- WinAVR install
- USB Driver install
- Example programs
- Atmel AVR ATmega168 microcontroller data sheet

For more details on your iRobot Command Module and to download the Owner's Manual, visit **www.irobot.com/create.**

Anatomy



The tables below show the signal connections for the each of the ePorts. The four ePorts have similar electrical connections, allowing add-on modules to be installed in different ePorts depending upon your needs.

Top Left ePort

Pin#	Description	Name
1	Analog Input 7	ADC7
2	Digital I/O (Port C pin 3) or Analog Input 4	PC3/ADC4
3	Digital I/O (Port B pin 3)	PB3
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Not Connected	NC
9	Low Side Driver 0	LD0

Top Right ePort

Pin#	Description	Name
1	Analog Input 6	ADC6
2	Digital I/O (Port C pin 2) or Analog Input 2	PC2/ADC2
3	Digital I/O (Port B pin 2)	PB2
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Not Connected	NC
9	Low Side Driver 0	LD0

Top Center ePort

Pin#	Description	Name
1	Digital I/O (Port C pin 5) or Analog Input 5	PC5/ADC5
2	Digital I/O (Port C pin 1) or Analog Input 1	PC1/ADC1
3	Digital I/O (Port B pin 1)	PB1
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Not Connected	NC
9	Low Side Driver 0	LD0

Cargo Bay ePort

Pin#	Description	Name
1	Digital I/O (Port C pin 4) or Analog Input 4	PC4/ADC4
2	Digital I/O (Port C pin 0) or Analog Input 0	PC0/ADC0
3	Digital I/O (Port B pin 0)	PB0
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Low Side Driver 1	LD1
9	Low Side Driver 2	LD2

Processor I/O Pins

The Command Module's Atmel AVR ATmega 168 microcontroller has 23 I/O pins, arranged into 3 ports (B, C and D), plus 2 additional analog input pins. The table below gives a description of each pin along with the location of the hardware to which it is connected. For information on how to control and read from these pins, see the Command Module Owner's Manual.

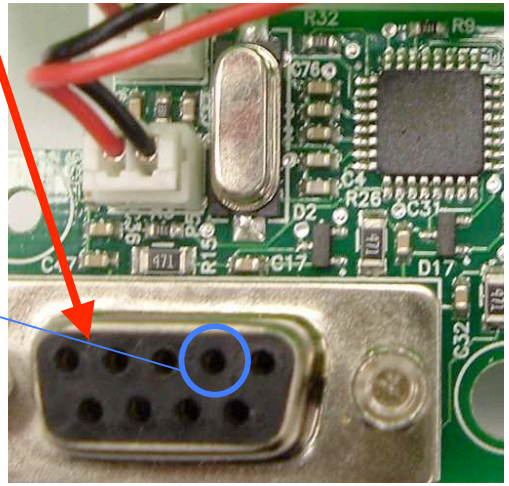
Pin Name	Description	Location
PB0	Digital I/O	Cargo Bay ePort pin 3
PB1	Digital I/O	Top Center ePort pin 3
PB2	Digital I/O	Top Right ePort pin 2
PB3	Digital I/O	Top Left ePort pin 3
PB4	Serial port connector select. 1 = USB port. 0 = iRobot Create	Internal
PB5	iRobot Create Power Detect. High if iRobot Create is on.	iRobot Create connector pins 10-13
PB6	Clock line	Internal Use only
PB7	Clock line	Internal Use only
PC0	Digital I/O or Analog Input	Cargo Bay ePort pin 2
PC1	Digital I/O or Analog Input	Top Center ePort pin 2
PC2	Digital I/O or Analog Input	Top Right ePort pin 2
PC3	Digital I/O or Analog Input	Top Left ePort pin 2
PC4	Digital I/O or Analog Input	Cargo Bay ePort pin 1
PC5	Digital I/O or Analog Input	Top Center ePort pin 2
PC6	Reset Line	Internal Use only
PD0	Serial Rx	iRobot Create connector pin 2 or USB
PD1	Serial Tx	iRobot Create connector pin 1 or USB
PD2	iRobot Create Device Detect Input	iRobot Create connector pin 15
PD3	USB Detect	USB port
PD4	Command Module Soft Button	Left button
PD5	Command Module LED 1	Left green LED
PD6	Command Module LED 2	Right green LED
PD7	iRobot Create Power Toggle (on rising edge)	iRobot Create connector pin 3
ADC6	Analog Input	Top Right ePort pin 1
ADC7	Analog Input	Top Left ePort pin 1

System Requirements:
Windows XP*
USB connection

*Support for additional operating systems may be available at **www.irobot.com/create.**

Top Left ePort

Pin#	Description	Name
1	Analog Input 7	ADC7
2	Digital I/O (Port C pin 3) or Analog Input 4	PC3/ADC4
3	Digital I/O (Port B pin 3)	PB3
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Not Connected	NC
9	Low Side Driver 0	LD0



Cargo Bay ePort

Pin#	Description	Name
1	Digital I/O (Port C pin 4) or Analog Input 4	PC4/ADC4
2	Digital I/O (Port C pin 0) or Analog Input 0	PC0/ADC0
3	Digital I/O (Port B pin 0)	PB0
4	Regulated 5V voltage (when iRobot Create is on)	Vcc
5	iRobot Create Battery Ground	Gnd
6	Not Connected	NC
7	iRobot Create Battery Voltage (when iRobot Create is on)	Vpwr
8	Low Side Driver 1	LD1
9	Low Side Driver 2	LD2

Introduction to Embedded Systems

Isaac Liu & Edward A. Lee & Sanjit Seshia

UC Berkeley

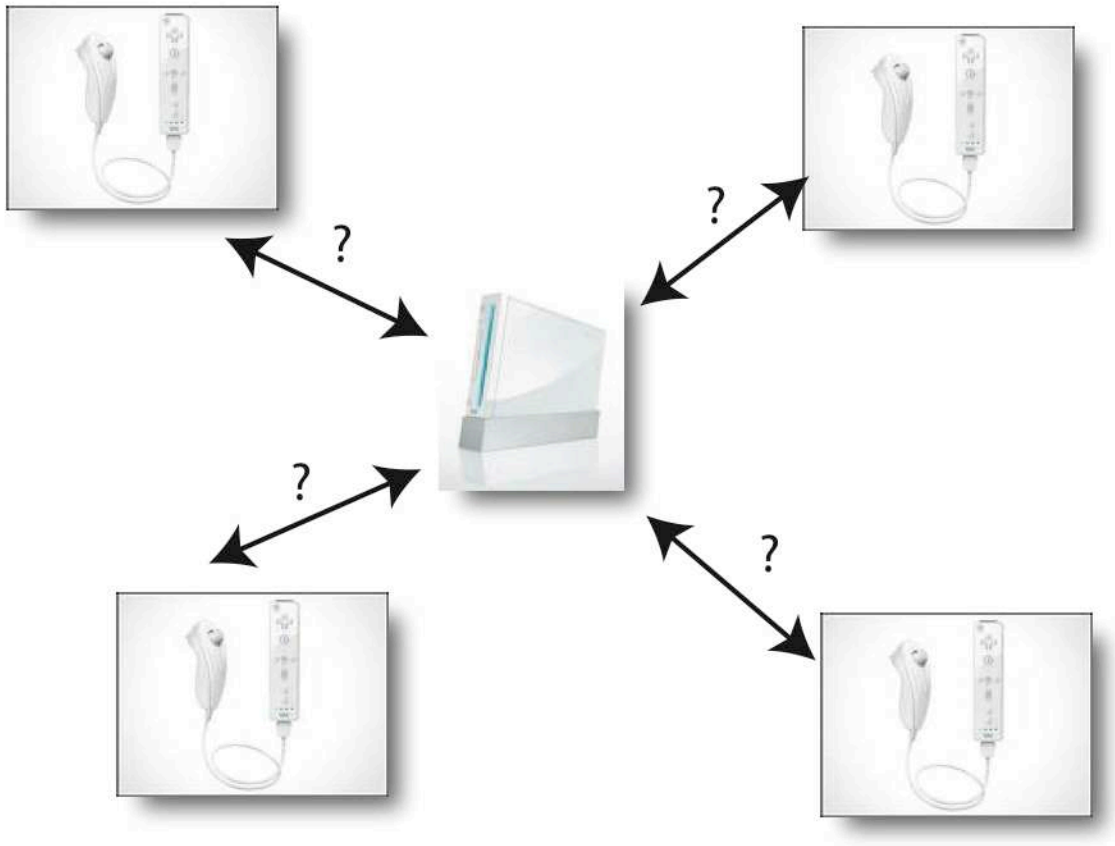
EECS 124

Spring 2008

Lecture 4: Interfacing to Sensors and Actuators

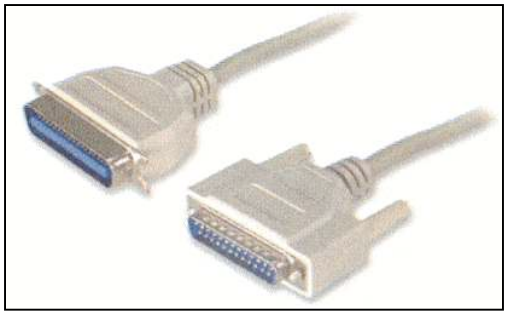
Communication

- Physical Connection
- Hand Shake
- Multiple connections
- Timing Characteristics



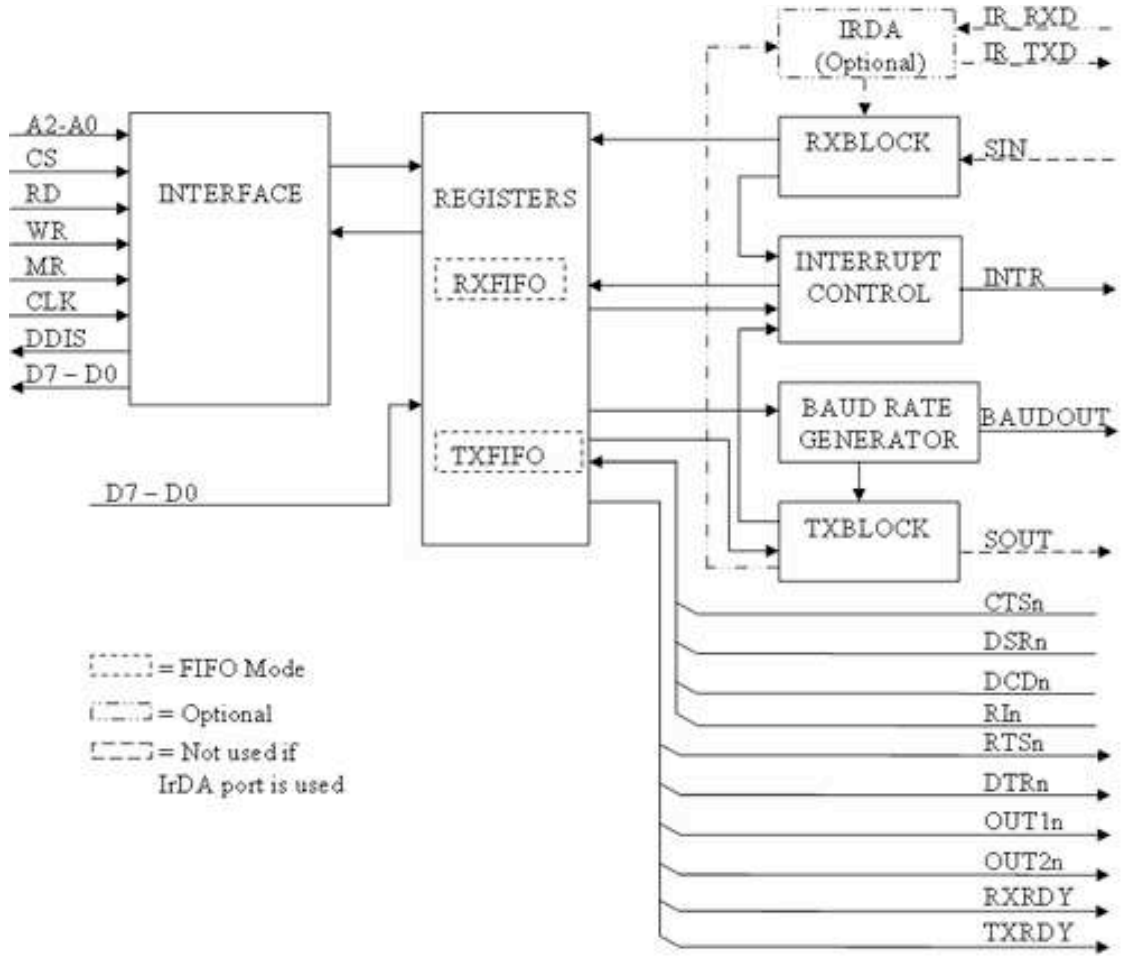
Parallel v.s Serial

- Parallel
  - Multiple data lines transmitting data
  - Speed, Simple hardware
  - Ex: PCI, ATA, CF cards, Bus
- Serial
  - Single data line transmitting data
  - Low Power, length
  - Ex: USB, SATA, SD cards,
    PCI-Express



Universal Asynchronous Receiver-Transmitter

- Convert serial data to parallel data, and vise versa.
- Uses shift registers to load store data
- Can raise interrupt when data is ready
- Commonly used with RS-232 interface



Standards

- Serial:
  - Synchronous:
    - SPI, I2C, JTAG, USB
  - Asynchronous:
    - RS232

- Parallel:
  - Bus protocols, Advanced Technology Attachment (ATA), Peripheral Component Interface (PCI)

Embedded Microcontroller

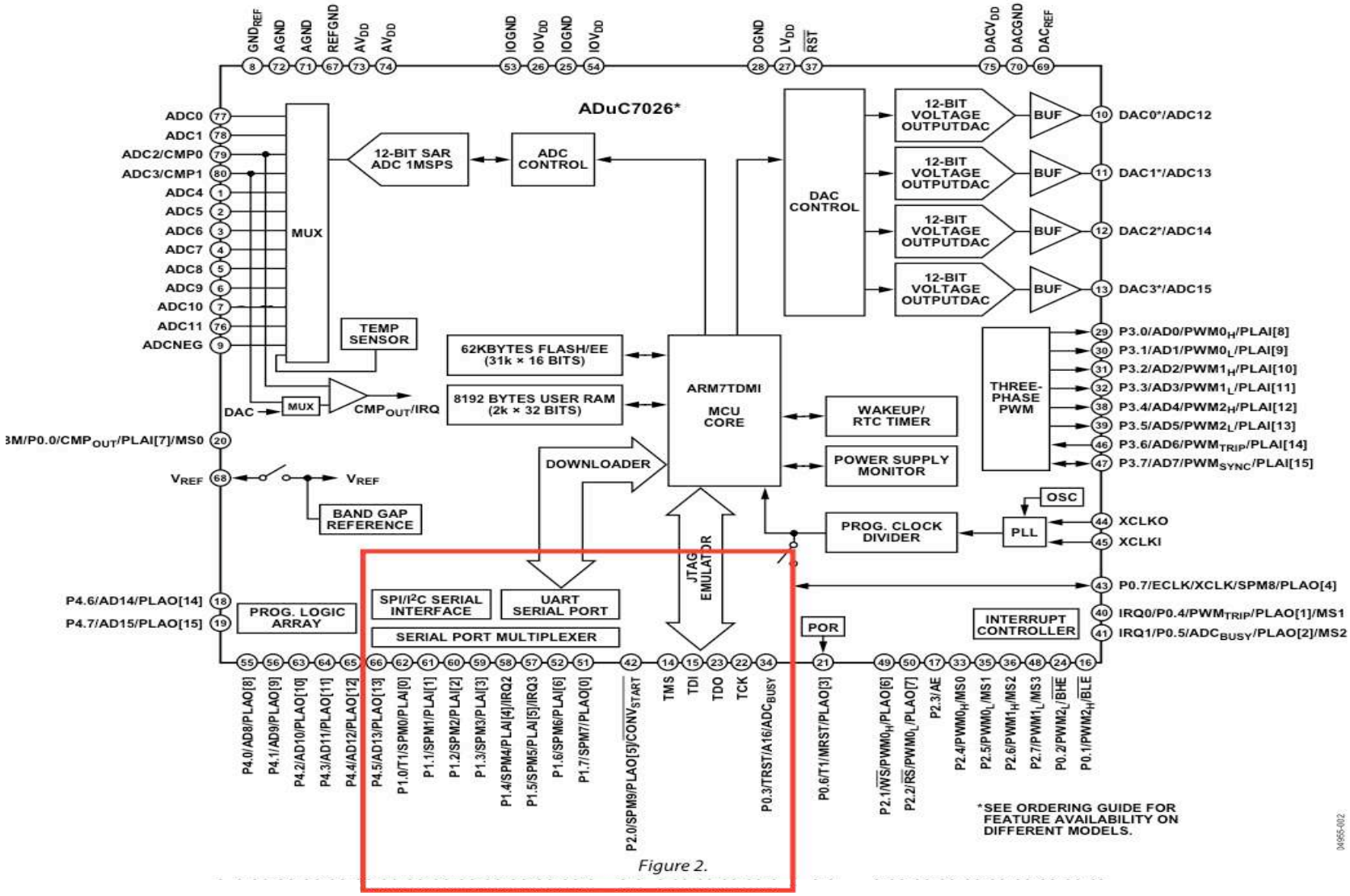
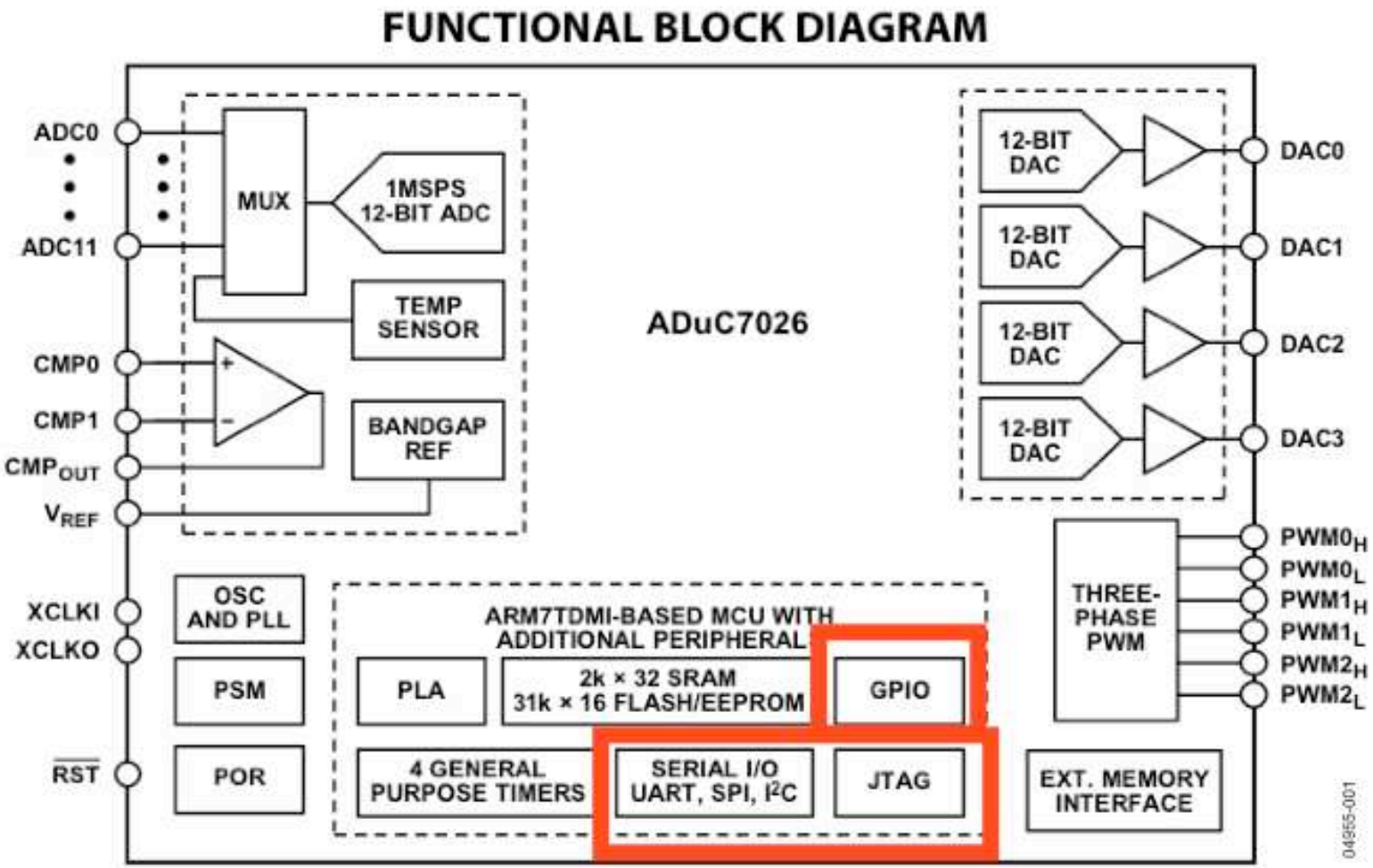


Figure 2.

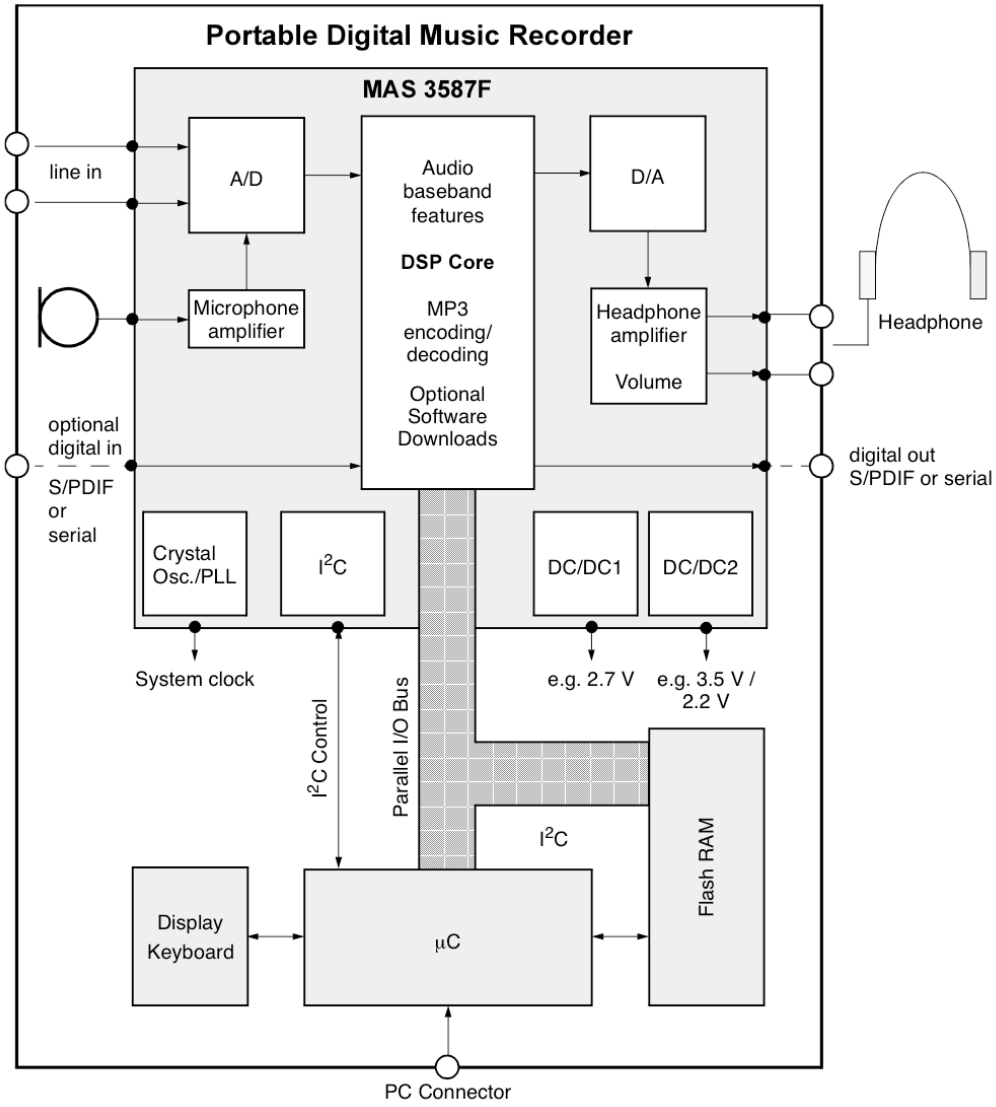
Embedded Microcontroller



Application

Simple MP3 Player

- Uses serial for control
- Parallel for data transfer
- Direct memory access



Basic C Review

Data Types:
(unsigned) int - 32 bits
(unsigned) short - 16 bits
(unsigned) char -8 bits

Bitwise Operators:

>>	Shift right
<<	Shift Left
&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR

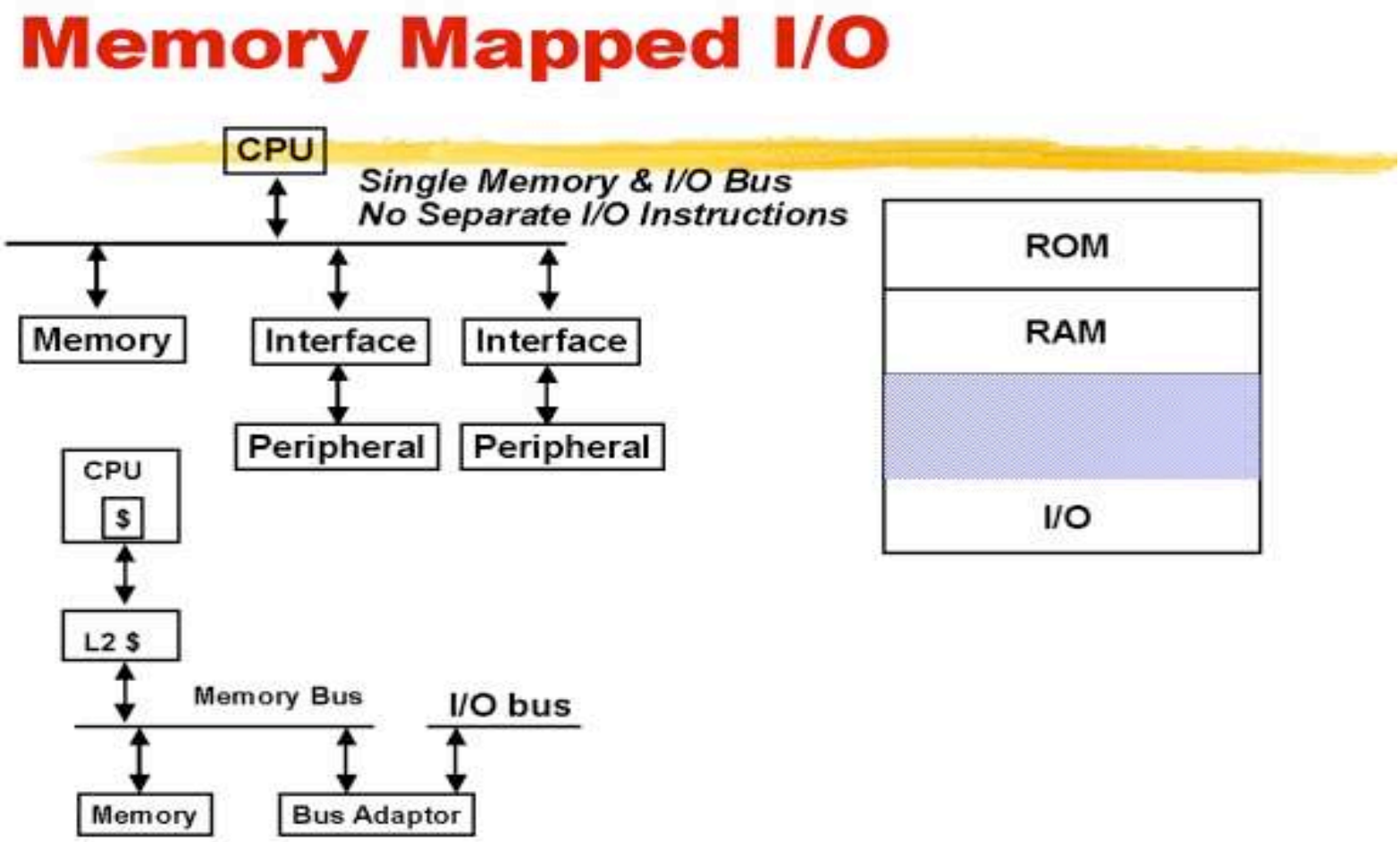
```
Pointers:
int* regaddr;  // regaddr is an address
int reg = 3;    // reg is a value
int tmp;

regaddr = &reg;  //Assigning regaddr to
                 //point to reg
// *regaddr is dereferencing regaddr

tmp = regaddr;   //What will tmp be?
tmp = *regaddr;  //What will tmp be?
                 //What will reg be?
*regaddr = 5;    //What will tmp be?
                 //What will reg be?
```

Questions?

Embedded Software Overview



Embedded Software Overview

```
#define LEDBothOff       (PORTD |= LEDBoth)
#define CmdStart        128
#define CmdControl      130
#define CmdFull         132
#define RadStraight     32768
```

```
int main (void)
{
  // Set up Create and module
  initialize();
  LEDBothOff;
  powerOnRobot();
  byteTx(CmdStart);
  baud(Baud28800);
  byteTx(CmdControl);
  byteTx(CmdFull);

  // Stop just as a precaution
  drive(0, RadStraight);

  //Main Loop
}
```

```
// Transmit a byte over the serial port
void byteTx(uint8_t value)
{
  while(!(UCSR0A & _BV(UDRE0))) ;
  UDR0 = value;
}
```

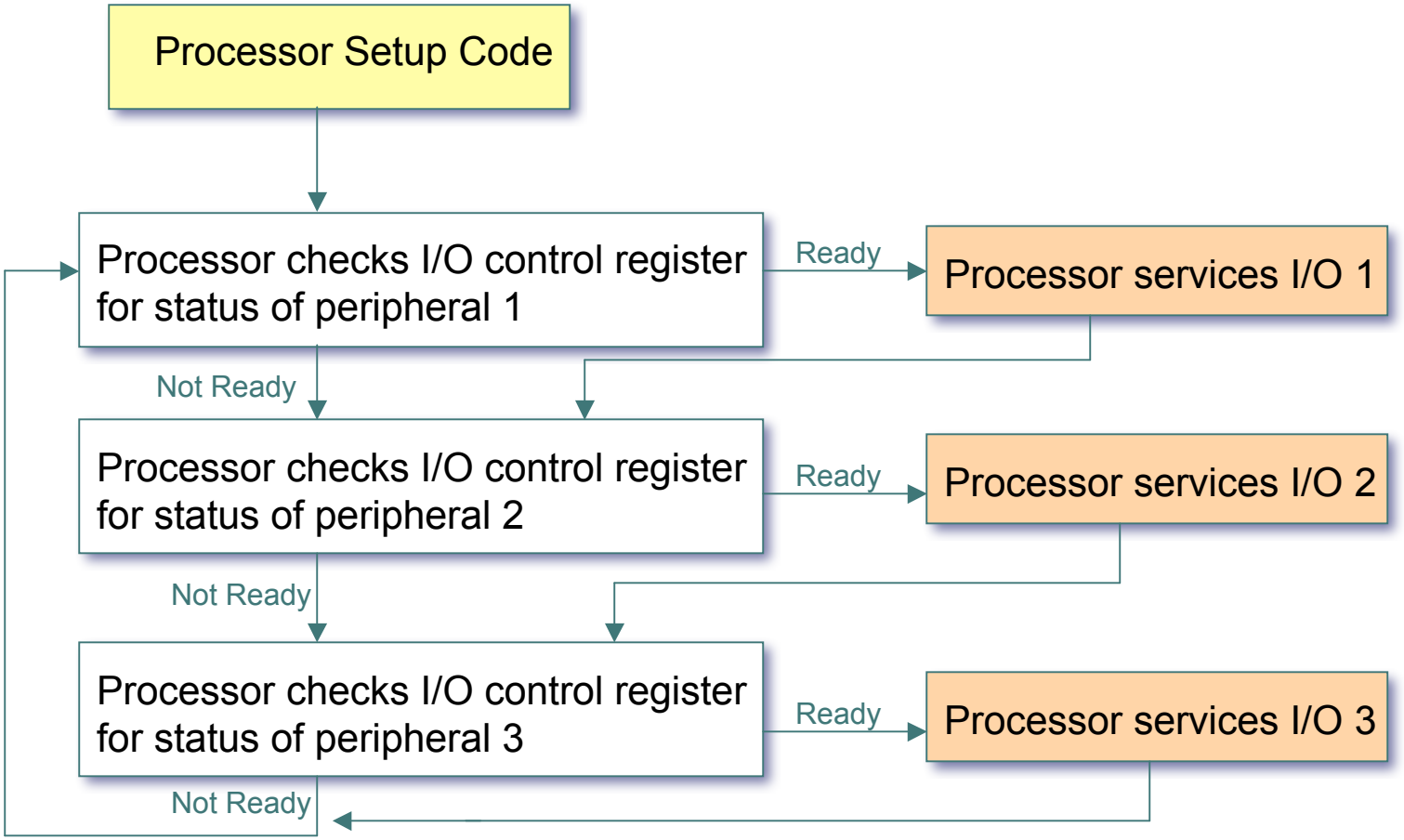
```
// Send Create drive commands in
//terms of velocity and radius
void drive(int16_t velocity, int16_t radius)
{
  byteTx(CmdDrive);
  byteTx((uint8_t)((velocity >> 8) & 0x00FF));
  byteTx((uint8_t)(velocity & 0x00FF));
  byteTx((uint8_t)((radius >> 8) & 0x00FF));
  byteTx((uint8_t)(radius & 0x00FF));
}
```

Multitasking

Embedded Software Architectures

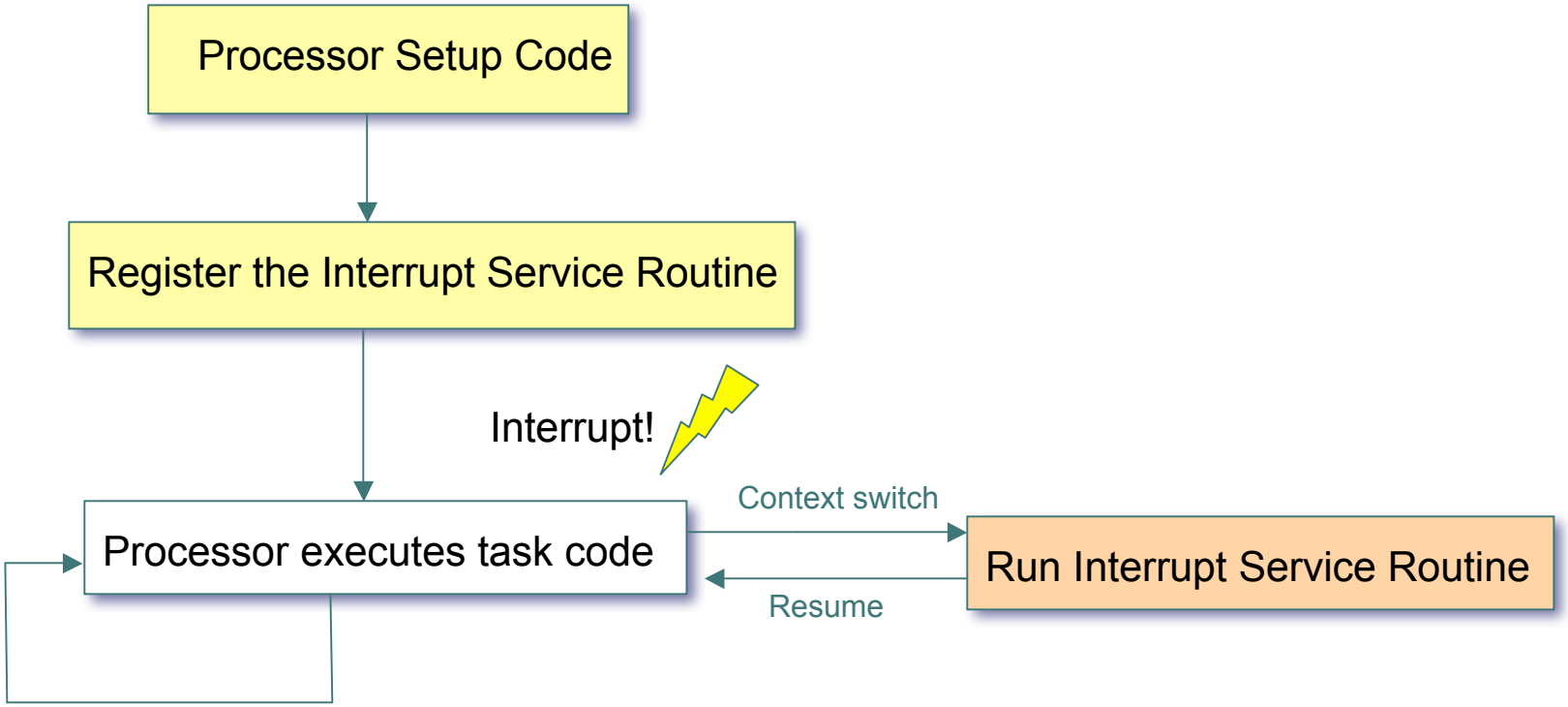
- Polling
  - Main loop checks each I/O device in turn and service any that need service.
  - Processor initiates communication
  - Simple, no overhead
  - Example: Web applications
- Interrupts
  - System reacts to triggers and then services the I/O
  - Peripheral initiates communication
  - Concurrency, sample rate
  - Example: UART, keyboard, timers

Polling


Processor Setup Code
Processor checks I/O control register for status of peripheral 1
Ready
Processor services I/O 1
Not Ready
Processor checks I/O control register for status of peripheral 2
Ready
Processor services I/O 2
Not Ready
Processor checks I/O control register for status of peripheral 3
Ready
Processor services I/O 3
Not Ready

Interrupts

- Interrupt Service Routine
  - Short subroutine that handles the interrupt



Interrupt example

```
static int iTemperatures[2];
void interrupt vReadTemperatures (void)
{
        iTemperatures[0] = //Read value from hardware 1
        iTemperatures[1] = //Read value from hardware 2
}
void main(void)
{
        int iTemp0, iTemp1;
        //Setup code

        while(TRUE)
        {
                iTemp0 = iTemperatures[0];
                iTemp1 = iTemperatures[1];
                if ( iTemp0 != iTemp1 )
                        // Set off alarm!
        }
}
```

Compiler can optimize this!!

What if interrupt updated both values here?

Shared Data

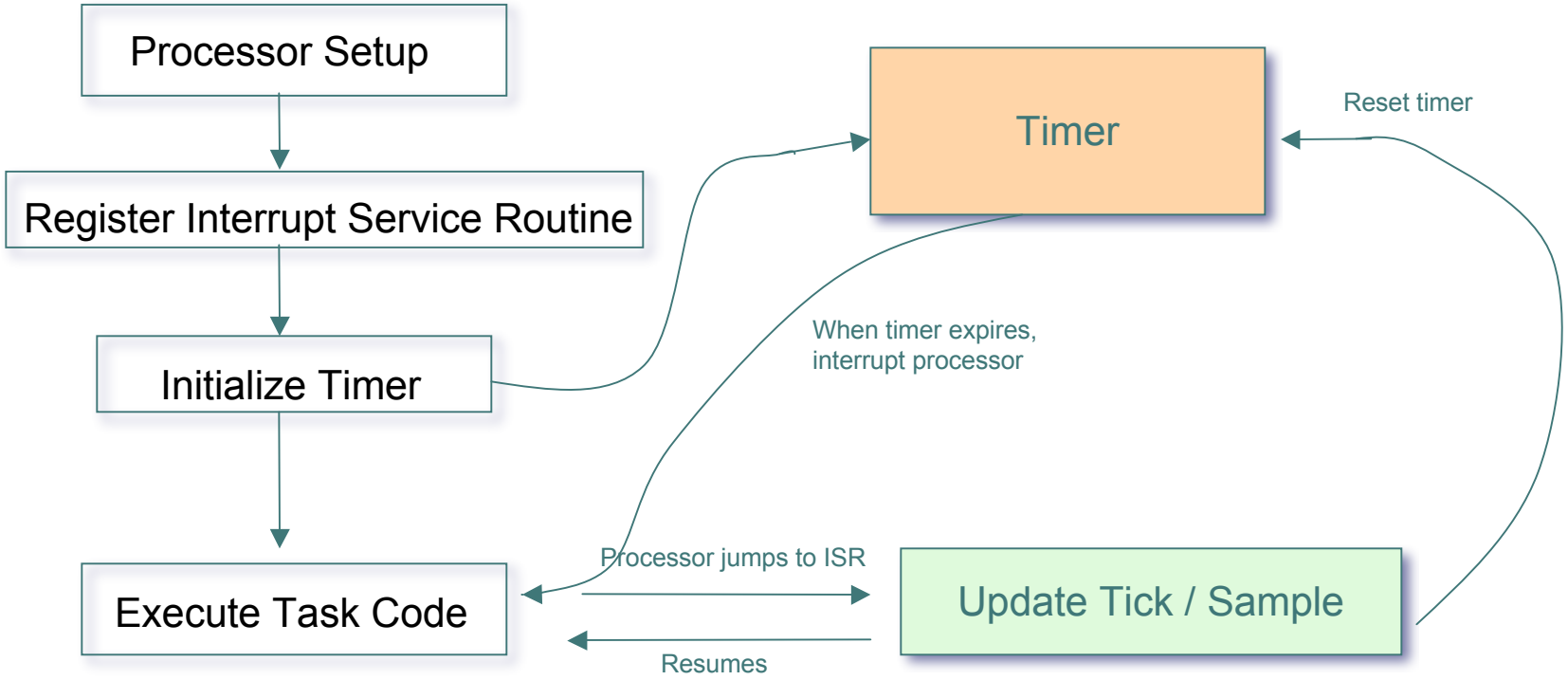
- Data consistency
  - Critical Section
    - Need to protect portion of the code from other access
    - Disable interrupts
  - Compiler Optimizations
    - Various optimization techniques
    - Volatile keyword, or turn off optimizations

Interrupt example revisited

```
static volatile int iTemperatures[2];
void interrupt vReadTemperatures (void)
{
        iTemperatures[0] = //Read value from hardware 1
        iTemperatures[1] = //Read value from hardware 2
}
void main(void)
{
        int iTemp0, iTemp1;
        //Setup code

        while(TRUE)
        {
                disableInterrupts();
                iTemp0 = iTemperatures[0];
                iTemp1 = iTemperatures[1];
                enableInterrupts();
                if ( iTemp0 != iTemp1 )
                        // Set off alarm!
        }
}
```

Timed Interrupt



```
#define _MMIO_BYTE(mem_addr) (*(volatile uint8_t *)(mem_addr))
#define _SFR_IO8(io_addr) _MMIO_BYTE((io_addr) + 0x20)
#define _SFR_MEM8(mem_addr) _MMIO_BYTE(mem_addr)
#define _BV(bit) (1 << (bit))
```

```
//Timer defines (iomx8.h)
#define TCCR1A  _SFR_MEM8 (0x80)
#define TCCR1B  _SFR_MEM8 (0x81)
/* TCCR1B */
#define WGM12   3
#define CS12    2
```

```
 //Enable interrupts (interrupt.h)
# define sei()  __asm__ __volatile__ ("sei" ::)
//Disable interrupts (interrupt.h)
# define cli()  __asm__ __volatile__ ("cli" ::)
#define SIGNAL(signame)                                 \
void signame (void) __attribute__ ((signal)); \
void signame (void)
```

SEI		Global Interrupt Enable
CLI		Global Interrupt Disable

```
void initialize(void)
{
  cli();

  // Set I/O pins
  DDRB = 0x10;
  PORTB = 0xCF;
  …….

  // Set up timer 1 to generate an interrupt every 1 ms
  TCCR1A = 0x00;
  TCCR1B = (_BV(WGM12) | _BV(CS12));
  OCR1A = 71;
  TIMSK1 = _BV(OCIE1A);

  // Set up the serial port with rx interrupt
  …….

  // Turn on interrupts
  sei();
}
```

```
// Global variables
volatile uint16_t timer_cnt = 0;
volatile uint8_t timer_on = 0;

// Timer 1 interrupt to time delays in ms
SIGNAL(SIG_OUTPUT_COMPARE1A)
{
  if(timer_cnt)
    timer_cnt--;
  else
    timer_on = 0;
}
```

```
void delayMs(uint16_t time_ms)
{
  timer_on = 1;
  timer_cnt = time_ms;
  while(timer_on) ;
}
```

Interrupt Latency

How much time it takes for a system to respond to an interrupt.

- Interrupt service routine execution time
- Context switch
- Higher priority interrupts
- Scheduling

iRobot Drive example

```
SIGNAL(SIG_USART_RECV) {
  uint8_t temp;
  temp = UDR0;

  if(sensors_flag) {
    sensors_in[sensors_index++] = temp;
    if(sensors_index >= Sen6Size)
      sensors_flag = 0;
  }
}
```

```
void delayAndUpdateSensors(uint16_t time_ms){
  uint8_t temp;

  timer_on = 1;
  timer_cnt = time_ms;
  while(timer_on) {
    if(!sensors_flag){
      for(temp = 0; temp < Sen6Size; temp++)
        sensors[temp] = sensors_in[temp];
      // Update running totals of distance and angle

      byteTx(CmdSensors);
      byteTx(6);
      sensors_index = 0;
      sensors_flag = 1;
    }
  }
}
```

```
for(;;) {
    delayAndUpdateSensors(10);
    if(UserButtonPressed)
    {
      // Drive around until a button or unsafe condition is detected
      while(!(UserButtonPressed) && (!sensors[SenCliffL])
&& (!sensors[SenCliffFL]) && (!sensors[SenCliffFR])
&& (!sensors[SenCliffR])&& (!sensors[SenChAvailable])){
       // Keep turning until the specified angle is reached
       if(turning)
       {  // Code to continue turning }

        // Check for a bump
       else if(sensors[SenBumpDrop] & BumpEither) {
         // Set the turn parameters and reset the angle
         if(sensors[SenBumpDrop] & BumpLeft)
           turn_dir = 0;
         else
           turn_dir = 1;
          //Command to turn iRobot }
       else {
         // Otherwise, drive straight
         drive(300, RadStraight); }

       // Flash the leds in sequence
       // Update LED State
       // wait a little more than one robot tick for sensors to update
       delayAndUpdateSensors(20);
     } //End while loop
     // Stop driving
     drive(0, RadStraight);
    }
}
```

References

1. Tammy Moergaard, Embedded Systems Architecture, A comprehensive guide for programmers, 2005
2. Stephen E. Derenzo, Practical Interfacing in the Laboratory, 2003
3. David E. Simon, An Embedded Software Primer, 1999
4. iRobot drive example, www.iRobot.com
5. AVR library, http://www.atmel.com/products/avr/