Solderless Breadboards

Power bus (red)
Ground bus (blue)
Component bus

Note that the two sides are not connected
Wiring Standards

When possible, use wire colors for different types of signals:

- Black: ground
- Red: power
- Other: various signals
Clean Wiring

A clean breadboard will make debugging easier – and it makes circuits more robust

www.linefollowing.com
tangentsoft.net
Care with Power

- Only insert components and wires into the breadboard when power is disconnected
- "Wire, check-twice, then power"
  - Never reverse power and ground (this is a very common mistake)
- Most chips that we will use expect +5V
  - More can destroy the chips
  - We will use DC/DC converters to step battery voltages down to +5V
Care of Chips

• Use insertion and extraction tools: never your fingers
• Minimize your contact with pins: static electricity can destroy a chip
• Use a wrist strap when you handle chips

www.a7vtroubleshooting.com
www.chantronics.com.au
Andrew H. Fagg: Embedded Real-Time Systems: Digital Practice
www.hvwtech.com
TTL Chips: 2-Input AND Gates

Chip number: 7408

Pin 1 is marked on the chip

Power

Ground

www.dcs.warwick.ac.uk

www2.117.ne.jp
TTL Chips: 2-Input OR/XOR Gates

7432 or 74LS32

7486 or 74LS86

www.dcs.warwick.ac.uk
TTL Chips: 3-Input AND Gates

7411

digikey.com
Constant Inputs

How do we configure a chip input as a constant?
Constant Inputs

How do we configure a chip input as a constant?

• For a constant 0: connect to ground
• For a constant 1: use a pull-up resistor to +5V (e.g., 10K ohm)
Wiring Procedure (Suggested)

- Power supply
- Power/ground buses
- Insert primary components
- Wire power/ground for components
- Add signals and remaining components
- Test incrementally
Debugging Techniques

• Multimeter:
  – Use *voltage mode* to check logic levels
  – Use *continuity mode* to confirm connections
    (but never with power turned on)

• Oscilloscope:
  – View voltage as a function of time on 2 channels

• Test incrementally

• Test intermediate sub-circuits
Debugging Techniques

Wire in LED to indicate logic level on a line

- For most components, do not allow the line to be driven by more than 20mA (check the specs if in doubt)

- Note that in this circuit, the LED turns on when logic level is LOW
Proposed Groups

Group 1:
• Hawkins
• Edwards*
• Hopkins
• ?

Group E:
• Watson
• Ritz
• Barajas Cortes
• Thompson
• Nicholas

Group 2:
• Littlefield
• Torres
• Goepfert
• ???

Group 3:
• Valentas
• Nakajima
• Sullivan
• Nelson

Group 4:
• Moerbeek
• Habib
• Murphy

Group 5:
• Striz
• Imai
• Lucas
• Bent
Today

• Finalize project groups
• Laboratory use details
• Project 1:
  – Specification
  – Initial steps
Administrivia

• Homework 2 due today @5:00
• As of today, we will have the lab open 18 hours per week
Group Assignments

Group 1:
- Hawkins
- Edwards*
- Hopkins
- Wood

Group 2:
- Littlefield
- Torres
- Goepfert
- Hickman

Group 3:
- Valentas
- Nakajima
- Sullivan
- Nelson

Group 4:
- Moerbeek
- Habib
- Murphy

Group 5:
- Striz
- Imai
- Lucas
- Bent

Group E:
- Watson
- Ritz
- Barajas Cortes
- Thompson
- Nicholas
Project 1: Beacon Tracker

- Robot is equipped with 4 infrared (IR) sensors
  - 2 facing forward
  - 2 mounted on a controllable turret
- 2 IR beacons in the environment

www.lynxmotion.com
Project 1: Beacon Tracker

Task:

• Robot starts by approximately facing one beacon
• Robot must turn to face beacon and then move toward it
• With forward motion, corrections may be necessary
• When the robot “sees” the second beacon to the left, the robot must stop
• Also: if no beacons are visible, the robot must also stop
System Overview

4 IR Sensors

Preprocessor

Your circuit

www.lynxmotion.com
Beacon Receiver

The preprocessor translates the IR sensor signal into a 2-bit number.

The state of each IR sensor is encoded with its own pair of bits:

<table>
<thead>
<tr>
<th>B1</th>
<th>B0</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>No signal</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Low signal</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Medium signal</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Strong signal</td>
</tr>
</tbody>
</table>
Robot Details: Control

Turret motor → Serial-to-PWM → Preprocessor

DC motor driver
Robot Control Interface

3 output lines will determine the motion of the robot.

<table>
<thead>
<tr>
<th>C2</th>
<th>C1</th>
<th>C0</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Stop</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Forward</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Backward</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Left</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Right</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Forward-Right</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Forward-Left</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>x</td>
</tr>
</tbody>
</table>
Robot Control Interface

2 output lines will determine the turret position

<table>
<thead>
<tr>
<th>T1</th>
<th>T0</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Forward</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Left</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Right</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>x</td>
</tr>
</tbody>
</table>
Your Job

Design and build a controller from basic logic gates

• Design the function: given each possible input from the sensors, what should the robot do?

• For each of these cases what command must you generate (C2, C1, C0, T1, T0)?

• What circuit will generate this command?

• Build the circuit
Hints

• A 7-chip *should* exist

• The preprocessor includes LEDs that enable you to see its inputs and outputs

• Do not underestimate the amount of time required to implement and debug your circuit
Hints

• Secure wires before running the robot

• Make sure that you connect batteries properly and that you bring power to your circuit
Power

We will use 2 batteries:
- 7.2V for the DC motors
- 9V for the control electronics
  – The preprocessor circuit will step this down to 5V and provide it to your circuit

- Never short power and ground!
- Make sure you place used batteries in the appropriate boxes for recharging
What You Turn In

By Thursday, February 23\textsuperscript{rd} (5:00pm):

- Demonstrate to me or Alois
- Project report:
  - Describe the function that you have implemented
  - K-Maps
  - Circuit design
- Personal report: rate the contribution of yourself and your lab-mates
Debugging/Safety Hints

• Start by testing your circuit prior to connecting motor power
• Once you connect motor power, put your robot up “on blocks” before running it on the floor
• Move a beacon around the robot to confirm that it performs appropriately
• Make sure you wire into your circuit the following rule:
  – If no beacon signal, then stop the motion of the robot
Lab Procedures

• No food or drink are allowed in the lab.

• Before leaving the lab, please be sure to clean up your workspace.

• Because some equipment may be in short supply, please coordinate with others who will need these resources.

• Never place dead components back into the stock (instead – place them in the ‘graveyard’).
Lab Procedures

• No equipment or supplies may leave the lab without the permission of the monitor.

• No books may leave the lab.

• Please clear all guests with the lab monitor.

• Unless you have prior permission, please do not handle the projects of other class members.
Lab Procedures

• Always check your wiring before you power up your circuit (especially your power and ground connections).

• When removing chips from breadboards, always use an appropriate tool (not your fingers!).

• If you break something, please report it (don't just put it away).

• You are expected to supply and configure your own laptop computers for project use.
Group Design (Now)

• What are the different possible sensor states?
• Can you simplify this set of states?
• A: For each state, what should the robot do?
• B: What is the truth table?
• C: Generate the K-maps

By end of class: hand in A, B & C
Schedule

• We currently have 3 robots up and running
  – groups will need to share
  – The robots are designed so that you will be able to easily remove your circuit while leaving the other components intact

• We will soon have all 5 robots up and running
Next Time

Sequential logic: time and memory

• Reading:
  – ESA 3.6.3 (sequential logic section)
  – D flip-flop discussion from playhookey.com
    • Focus on the inputs/outputs (not the gate-level implementation)