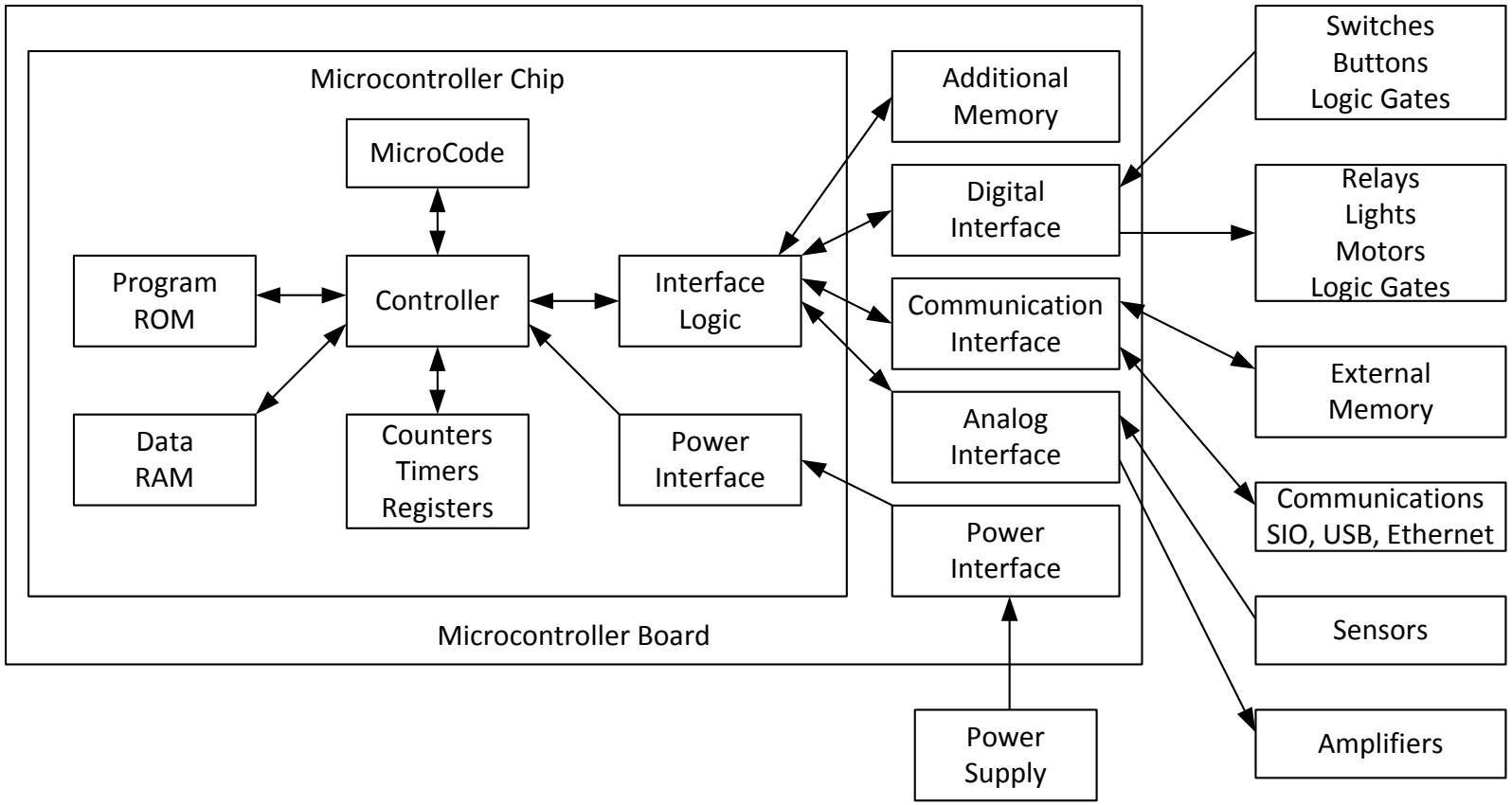
Microcontrollers for Ham Radio
Part 1 - BASIC Stamp

By Terry G. Glagowski / W1TR - 2012-03-22 23:17

- What is a Microcontroller?
- Some HAM Applications
- Popular Microcontrollers for Ham Radio
- The BASIC Stamp – Parallax
- Installing and Using the BASIC Stamp
- The Original BASIC Language
- The BASIC Stamp Integrated Development Environment
- Introduction to PBASIC Programming
- Some Example Programs
- Summary

What IS a Microcontroller?



Popular Microcontrollers for Ham Radio

- Arduino
- PIC
- Parallax Propeller
- BASIC Stamp (some versions use PIC chip)
- Older and Commercial Units
  - Z800 / Z80 / Z8 (Zilog) – Z8 had a BASIC interpreter
  - 8080 / 8085 / 8086 / 8088 / Pentium family (Intel/AMD)
  - 1802 (RCA)
  - 6800 / 68000 (Motorola)
  - 6502 (Rockwell)
  - TMS 9900 / TMS320Cxx DSP (Texas Instruments)
  - TLSC870 (Toshiba)

Some HAM Applications

- Automatic Antenna Tuner
- Automatic Screwdriver Antenna Controller
- Automatic Amplifier Controller / Tuner
- Multi Radio to Multi Antenna Switch Controller
- Transceiver Controller (in every radio)
- DDS Synthesizer (PIC-EL)
- CW Keyer – ARRL PIC Kit
- Rotor Controller
- ??? Use Your Imagination …

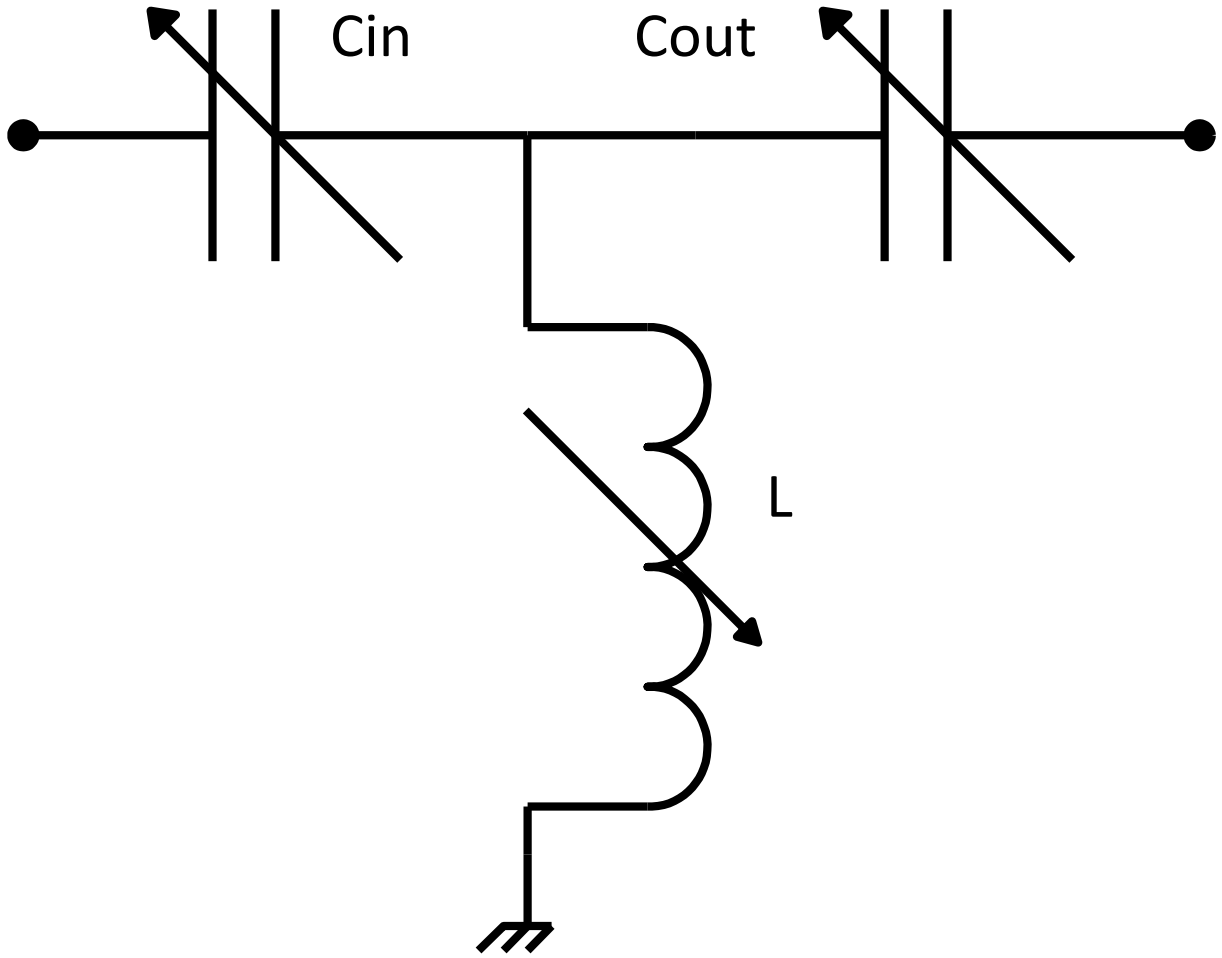
Update the Heathkit SA-2500 ATU

- Purchased the SA-2550 ATU in 1993
- HeathKit going out of business, deep discount
- Claim: automatic antenna tuner
- Reality:
  - Turn bandswitch to Set Coil Electrically, Comparator Circuit
  - Input and Output Capacitors Turn Randomly
  - Tuning Stops When SWR is Coincidentally Low Enough?
  - HUH? Not Really All That Automatic, It's STUPID !!!
- Since 1993, Thought About a Microcontroller
- Procrastination Set In, then Life Circumstances!
- Article: "Tuner Transformation", Peter Ferrand, 73 Magazine, April, 1987
  http://www.nostalgickitscentral.com/heath/73-index/articles/SA-2500-4-87.pdf

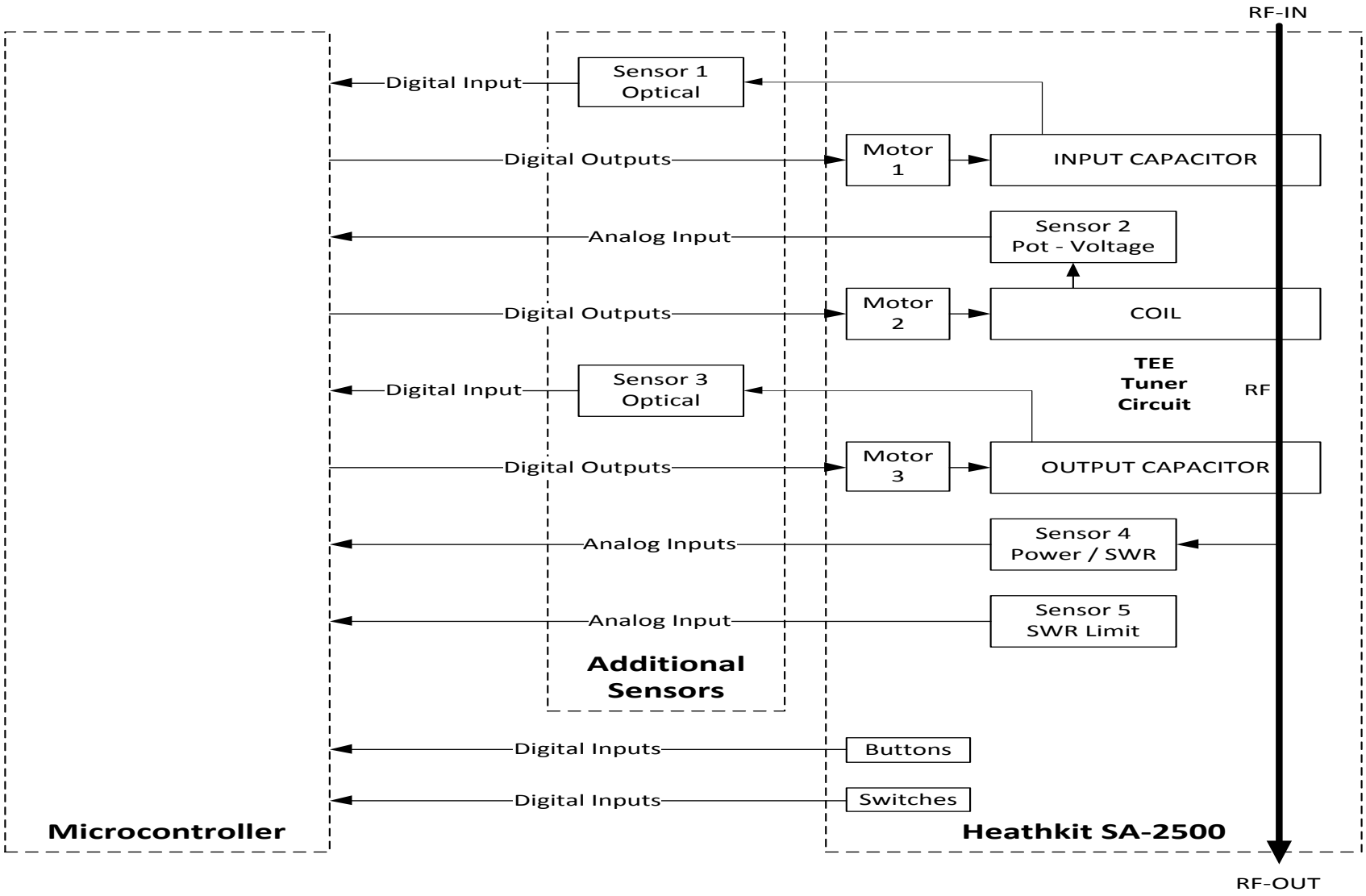
Heathkit SA-2500

SA-2500 Front Panel

Antenna Tuner “TEE” Circuit



Automatic Antenna Tuner



Sensor / Effector Signals

Automatic Antenna Tuner Sensor Signals.xlsx

	A	B	C	D	E
1	**Sensor**	**Signal**	**Type**	**Direction**	**Remarks**
2	Sensor 1	Input Capacitor Position	Digital	Input	Indicates Input Capacitor is at Home Position
3	Sensor 2	Coil Position	Analog	Input	Indicates Coil Position - Voltage from Potentiometer
4	Sensor 3	Output Capacitor Position	Digital	Input	Indicates Output Capacitor is at Home Position
5	Sensor 4	Forward Power	Analog	Input	Indicates Forward Power Level - Power Output
6	Sensor 4	Reflected Power	Analog	Input	Indicates Reflected Power Level
7	Sensor 5	SWR Limit	Analog	Input	Indicates Acceptable SWR Limit - Voltage from Potentiometer
8	Switch 1	Input Capacitor UP	Digital	Input	Center Off Switch UP Position - Increase Input Capacitance
9	Switch 1	Input Capacitor DOWN	Digital	Input	Center Off Switch DOWN Position - Decrease Input Capacitance
10	Switch 2	Coil UP	Digital	Input	Center Off Switch UP Position - Increase Inductance
11	Switch 2	Coil DOWN	Digital	Input	Center Off Switch DOWN Position - Decrease Inductance
12	Switch 3	Output Capacitor UP	Digital	Input	Center Off Switch UP Position - Increase Output Capacitance
13	Switch 3	Output Capacitor DOWN	Digital	Input	Center Off Switch DOWN Position - Decrease Output Capacitance
14	Switch 4	Band SW - Sig 1 thru Sig 10	Digital	Input	Indicates Band (i) is Selected (may need to do BCD encoding)
15	Switch 5	Band High / Low	Digital	Input	Indicates Band HI or LO is Selected
16	Button 1	Auto Tune	Digital	Inut	Activates / Deactivates Automatic Tune Cycle
17	**Effector**	**Signal**	**Type**	**Direction**	**Remarks**
18	Motor 1	Forward	Digital	Output	Controls Motor 1 to Move Forward
19	Motor 1	Reverse	Digital	Output	Controls Motor 1 to Move Reverse
20	Motor 2	Forward	Digital	Output	Controls Motor 2 to Move Forward
21	Motor 2	Reverse	Digital	Output	Controls Motor 2 to Move Reverse
22	Motor 3	Forward	Digital	Output	Controls Motor 3 to Move Forward
23	Motor 3	Reverse	Digital	Output	Controls Motor 3 to Move Reverse

Sheet1 Sheet2 Sheet3

Control Algorithm

- Always Move Input Cap, Coil, Output Cap Re Switches
- Begin Automatic When AutoTune Button Pressed
- Stop Automatic When AutoTune Button Pressed Again
- Read Band and Hi/Lo Switches, Preset Coil
- Iteratively Tune Output Cap, Input Cap, Coil
  While Reading Forward and Reflected Power
- Adjust Components for Maximum Forward / Reflected Ratio
- Stop at Minimum, Less Than Acceptable Limit, or Abort
- More-Or-Less How you would do it manually!

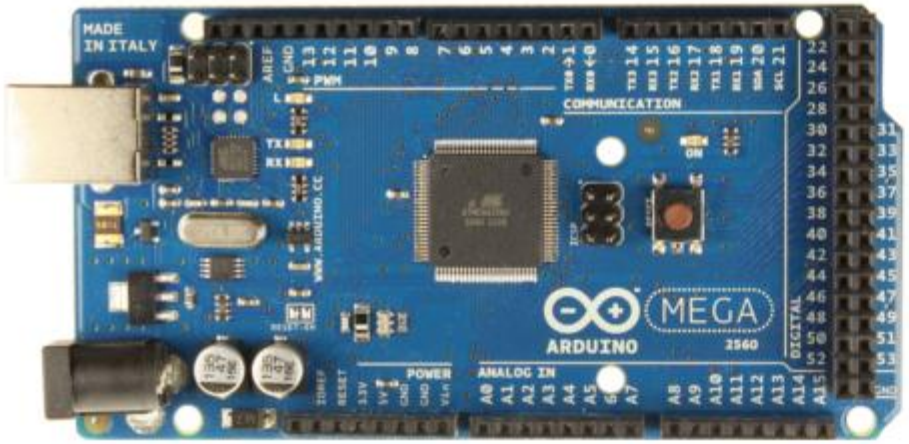
What Microcontroller to Use?

- Price
- Availability
- Ease of Programming
- Capabilities
  - Processing
  - I/O
  - Memory

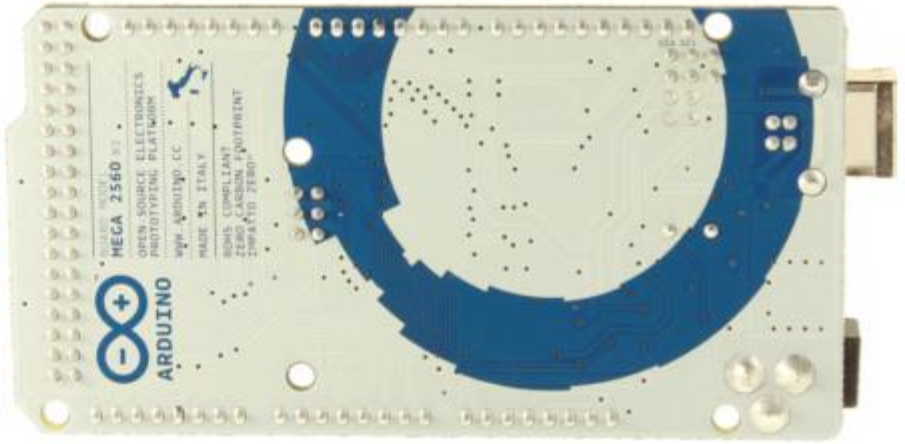
Arduino

http://arduino.cc/en/Main/ArduinoBoardMega2560

Arduino Mega 2560

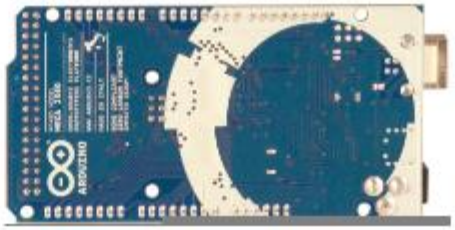


Arduino Mega 2560 R3 Front



Arduino Mega2560 R3 Back

Arduino Mega 2560 Front



Arduino Mega 2560 Back

Arduino Mega 2560 Capabilities

Summary

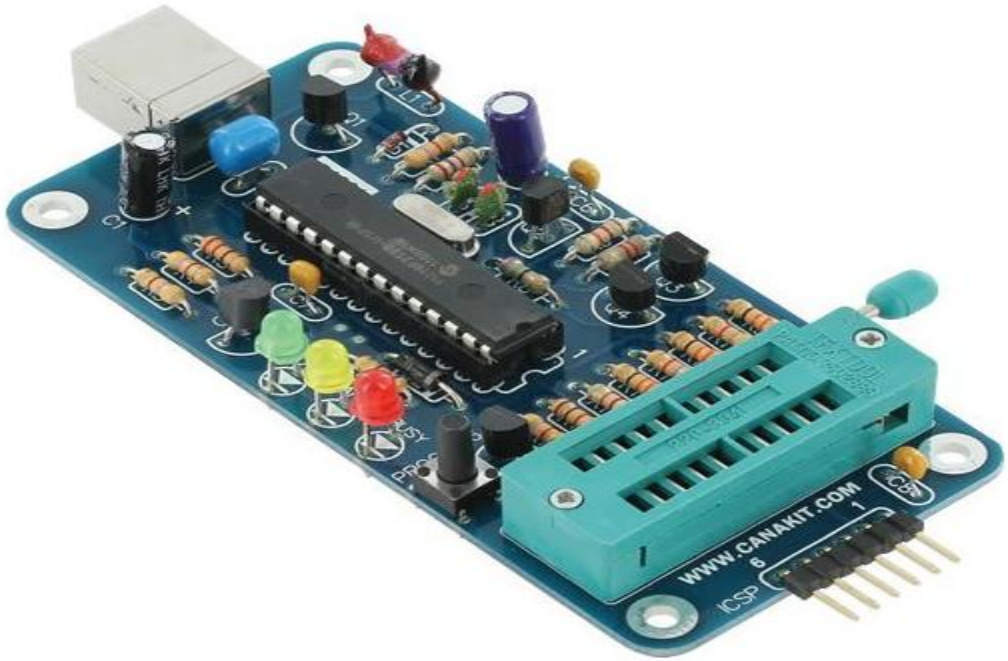
Programming Language	C / Assembly using Integ. Dev. Env.
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Programmable Interface Controller - PIC

www.microchip.com

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

- Originally Developed by General Instrument, 1975
- Just like McDonalds – over 10 billion sold!
- MANY variations of PIC chips – see data sheets on WEB



PIC Capabilities (8 bit)

- Bits: 8, 16, 32 versions
- Programming Language: MPLAB IDE - C, Assembly
- Operating Voltage: 5 VDC
- Memory: Flash, EEPROM 14KB, RAM 1024
- I/O Pins: 12
- Speed: 32 MHz
- Timers: (2) 8-bit, (1) 16 bit, Watchdog
- A/D: 8 ch, 10 bit
- Comparators: 2
- PWM 1
- USART 1
- MSSP 1

PIC Capabilities (16 bit)

- Bits: 8, 16, 32 versions
- Programming Language: MPLAB IDE - C, Assembly
- Operating Voltage: 3.6 Vdc
- Memory: Flash, EEPROM 256 b, RAM 512, ROM 4KB
- Speed: 8, 32 MHz
- 16 MIPS Performance
- Multiplier Unit: 16 x 16
- Divider Unit: 32 x 16
- Timers: (3)16 bit, Watchdog
- A/D: 7 ch, 10 bit, 500 ksps
- Comparators: 2
- PWM 1
- UART 1
- SPI 1
- I2C 1

Propeller Quick Start - Parallax

The Propeller Capabilities

- 32 bit 8 core multiprocessor P8X32A
- 160 MIPS
- 64KB EEPROM: 32KB Code, 32KB General
- 32 I/O pins
- Programming Language: SPIN, Assembly

The BASIC Stamp – Parallax

HomeWork Board – BS2 Module

http://www.parallax.com/go/wam

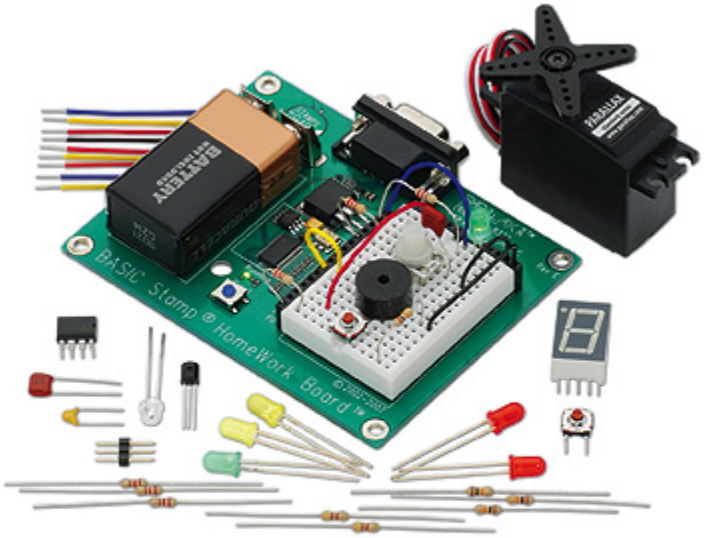
http://www.parallax.com/tabid/214/Default.aspx

Downloads & Resources	'Mini Projects'	Videos	Translations

What's a Microcontroller? is our first-step BASIC Stamp 2 tutorial, introducing the essentials electronics and programming side by side. You will write PBASIC programs and build both simple and advanced breadboard circuits with LEDs, pushbuttons, potentiometers, light sensor, a servo, and much more!

Check the complete components listing.

Audience: Anyone interested in getting started with microcontrollers and electronics! What's a Microcontroller? is the gateway to our **Stamps in Class program series** and is found in hobbyist's homes as well as in middle, high and vocational schools, colleges and universities in pre-engineering and basic electronic courses. The text is designed to accommodate a wide range of ages and skill levels and is a great resource for STEM programs.



Kit Choices:
The **What's a Microcontroller? text** and hardware is available in many different kit forms; the BASIC Stamp Discovery Kit features the BASIC Stamp 2 and Board of Education:

- BASIC Stamp Discovery Kit - Serial (With USB Adapter and Cable)
- BASIC Stamp Discovery Kit (USB)
- BASIC Stamp Activity Kit Serial + USB (Text v3.0)
- BASIC Stamp Activity Kit - Serial (Text v2.2)

BASIC Stamp Versions

Figure 1-2
BASIC Stamp 2 Microcontroller Module

In this text, "BASIC Stamp" refers to the BASIC Stamp® 2 microcontroller module. Designed and manufactured by Parallax Incorporated, this module's name is commonly abbreviated BS2, and it's the first in the series of modules shown in Figure 1-3. Each of the other modules is slightly different, featuring higher speed, more memory, additional functionality, or some combination of these extra features. To learn more, follow the "Compare BASIC Stamp Modules" link at www.parallax.com/basicstamp.

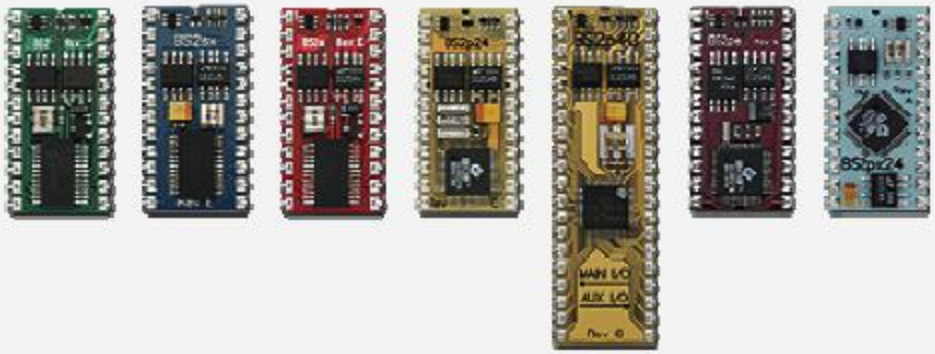


Figure 1-3
BASIC Stamp 2 Models, left to right: BS2, BS2e, BS2sx, BS2p24, BS2p40, BS2pe, BS2px

BASIC Stamp BS2 Capabilities

- Programmable in BASIC (PBASIC)
- Supports Windows, Mac, Linux
- First introduced in 1992
- CPU PIC16C57c 20 MHz, 4000 instr/sec
- RAM 32 Bytes, 6 I/O, 26 Variables
- EEPROM 2K Bytes 500 instructions
- I/O 16 bits bidirectional, + 2 SIO
  - Digital Output
  - Digital Input
  - Analog Output – Pulse Width Modulation (PWM)
  - Analog Input – read potentiometer position R/C network
  - Need A/D Chip or Comparator to read voltage input

The BASIC Language

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

- Beginners All-purpose Symbolic Instruction Code
- Invented at Dartmouth University in 1964
- By John Kemeny and Thomas Kurtz
- Designed to allow students to write programs easily
- For the Dartmouth Time Sharing System (DTSS)
- Used in mini computers and early home computers
  - Apple II
  - Comodore PET
  - Radio Shack TRS-80
  - S-100 Systems
  - IBM PC & XT
- BASIC is the “Trailor Trash” of the software programming world!
  It’s SIMPLE and IT WORKS!
- Computer Scientists DO NOT LIKE BASIC (or FORTRAN or Assembly Language)
  (Dykstra et.al. - structured programming: NO GOTOs, NO POINTERS)
- They prefer “structured languages” like ALGOL, Simula, Pascal, Modula
  and they barely tolerate “C” and “C++” but C# and Java seem acceptable!

Original BASIC Programming Statements

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

- DIM – defines data variables
- DATA – defines collection of data much like an input file but in memory
- MAT – defines a matrix (Was in Dartmouth BASIC, no longer in mainstream BASIC)
- LET – assigns expression to variable
- IF … THEN … ELSE – simple decision
- FOR … NEXT – simple loop with iteration
- WHILE … WEND – loop with termination condition
- DO … {WHILE} … {UNTIL} … LOOP – loop with termination condition
- GOTO – change of program sequence
- GOSUB – subroutine call
- RETURN – return from subroutine
- ON … GOTO / GOSUB – exception condition
- PRINT - output
- INPUT – input
- TAB or AT – position display cursor
- REM – remarks / comments for documenting code
- END – marks end of program
- PBASIC includes a lot more for embedded control & monitoring

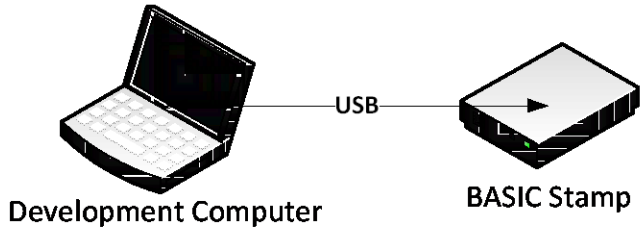
Installing and Using the BASIC Stamp

- Go To URL: http://www.parallax.com/
- Download and Install BASIC Stamp Software
  Including BASIC Stamp Editor and USB Drivers
- Put a 9 volt battery in the kit
- Connect the USB cable to the kit
- Connect the USB cable to the computer
- Operate the BASIC Stamp Editor

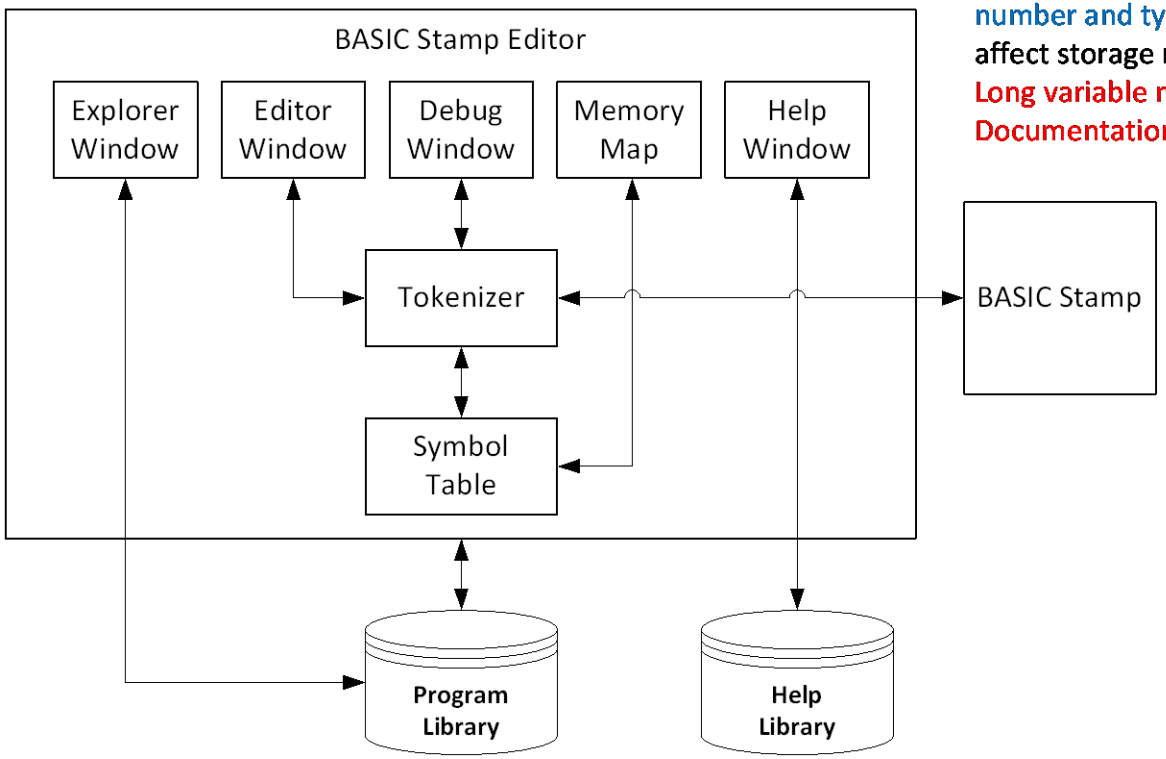
Install BASIC Stamp Software

http://www.parallax.com/BasicStampSoftware

Integrated Development Environment

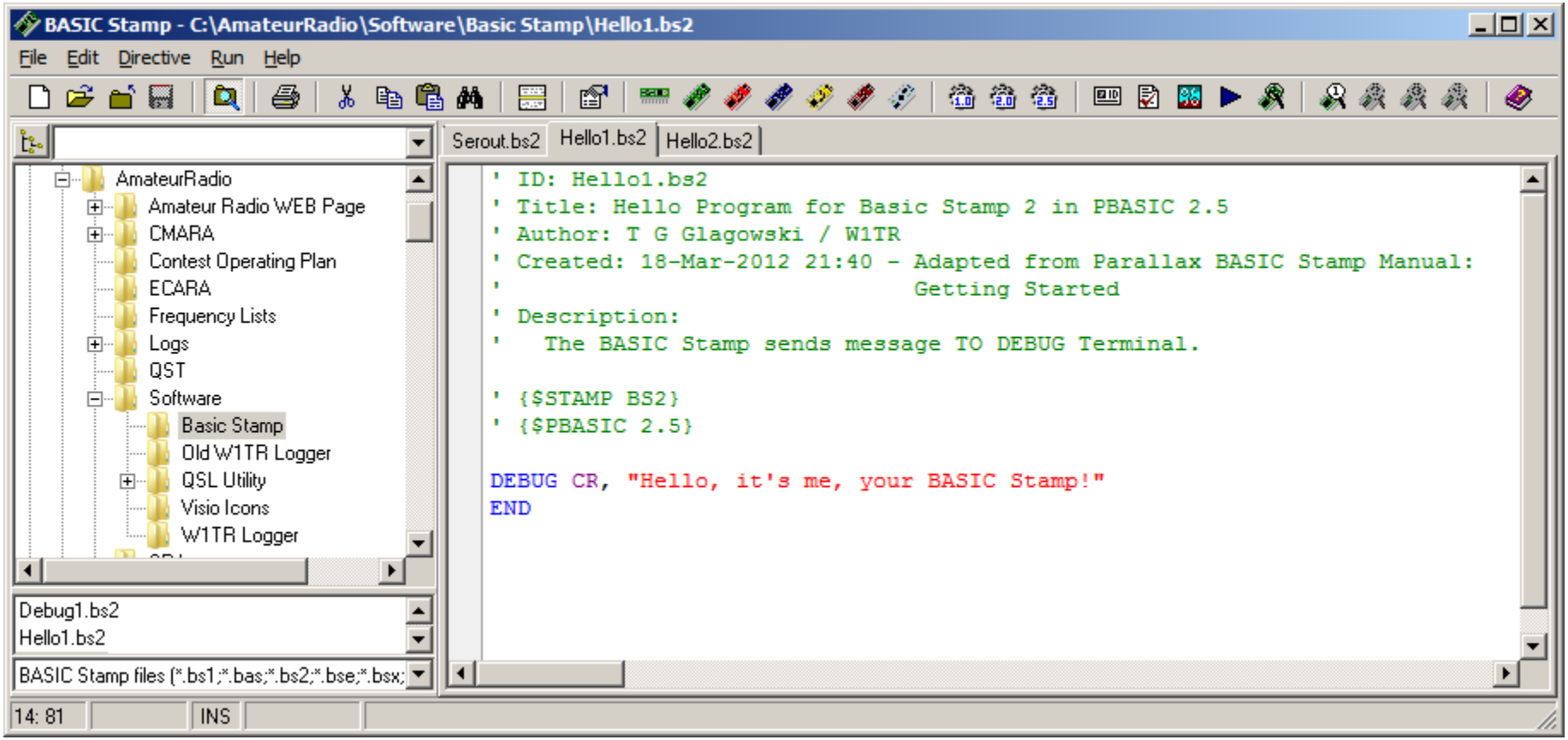


Tokenizer reduces storage requirements in Stamp.
Size of the source program does not matter,
number and size of variables,
number and type of executable statements
affect storage requirements.
Long variable names have no impact on storage required.
Documentation is stripped from the program code.

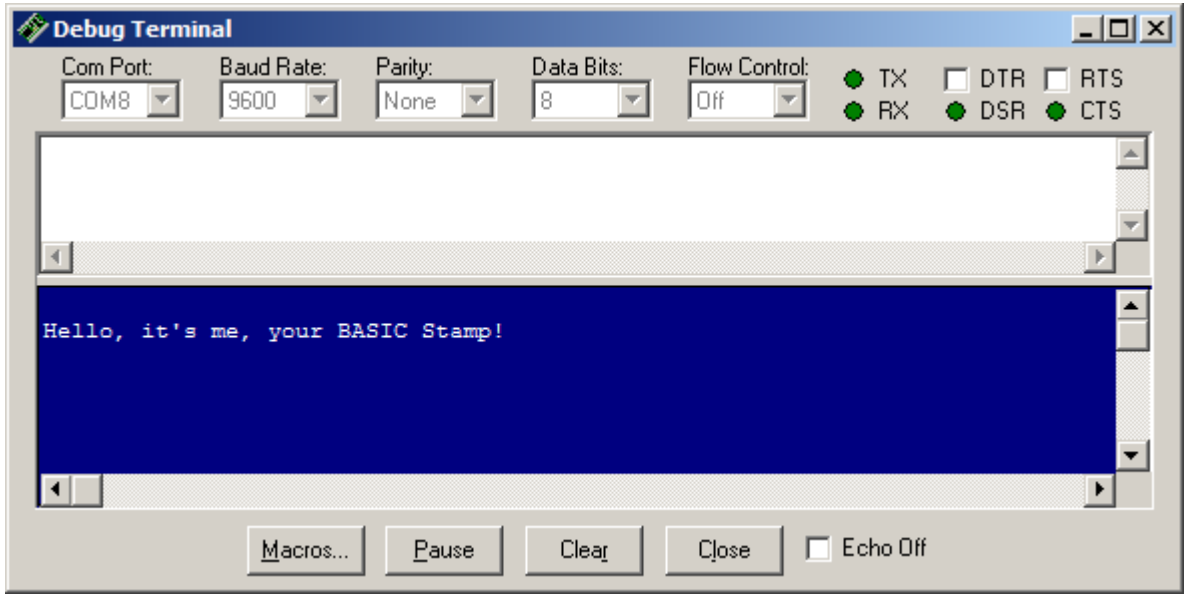


BASIC Stamp IDE Explorer & Editor Window

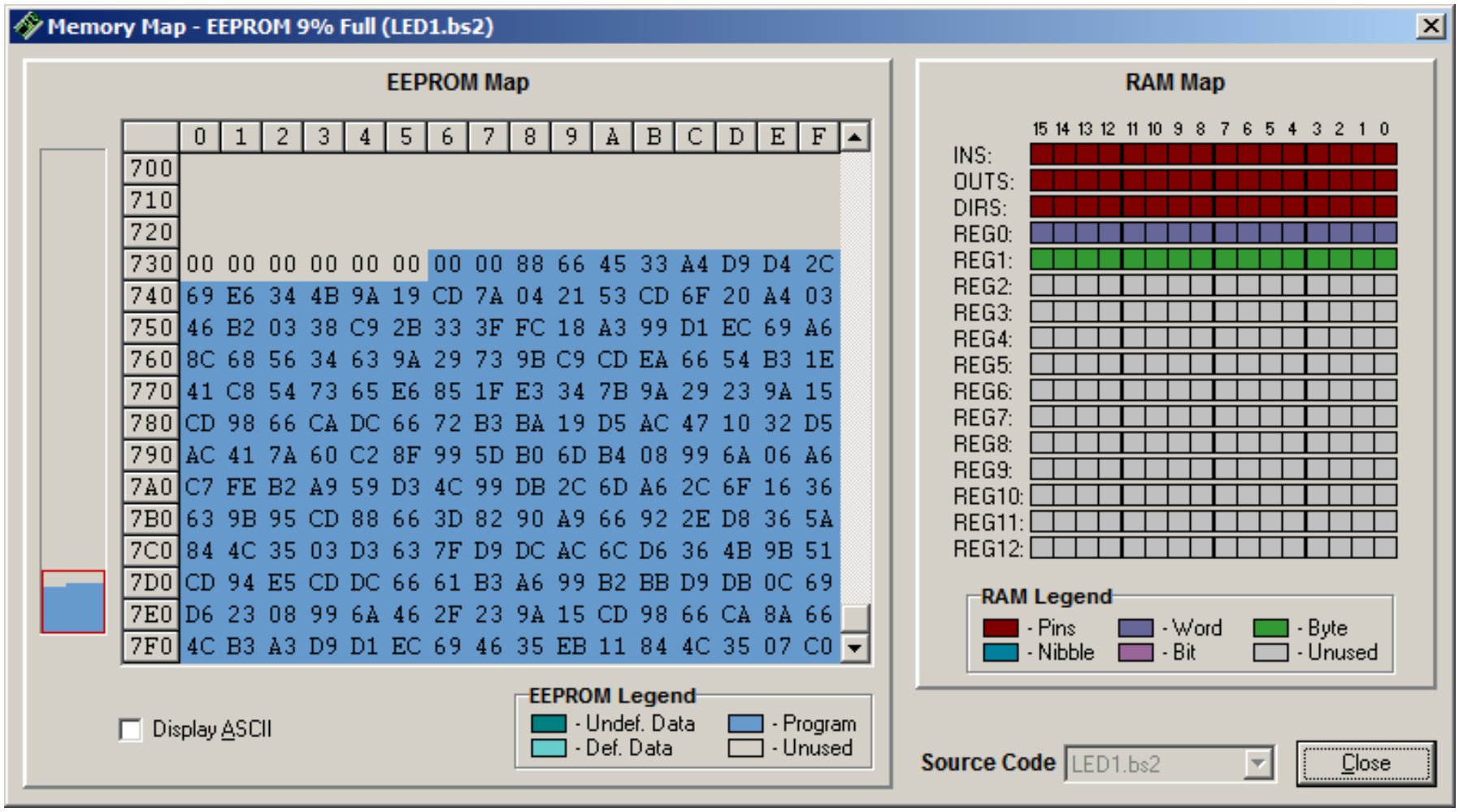
The HELLO Program for BASIC Stamp



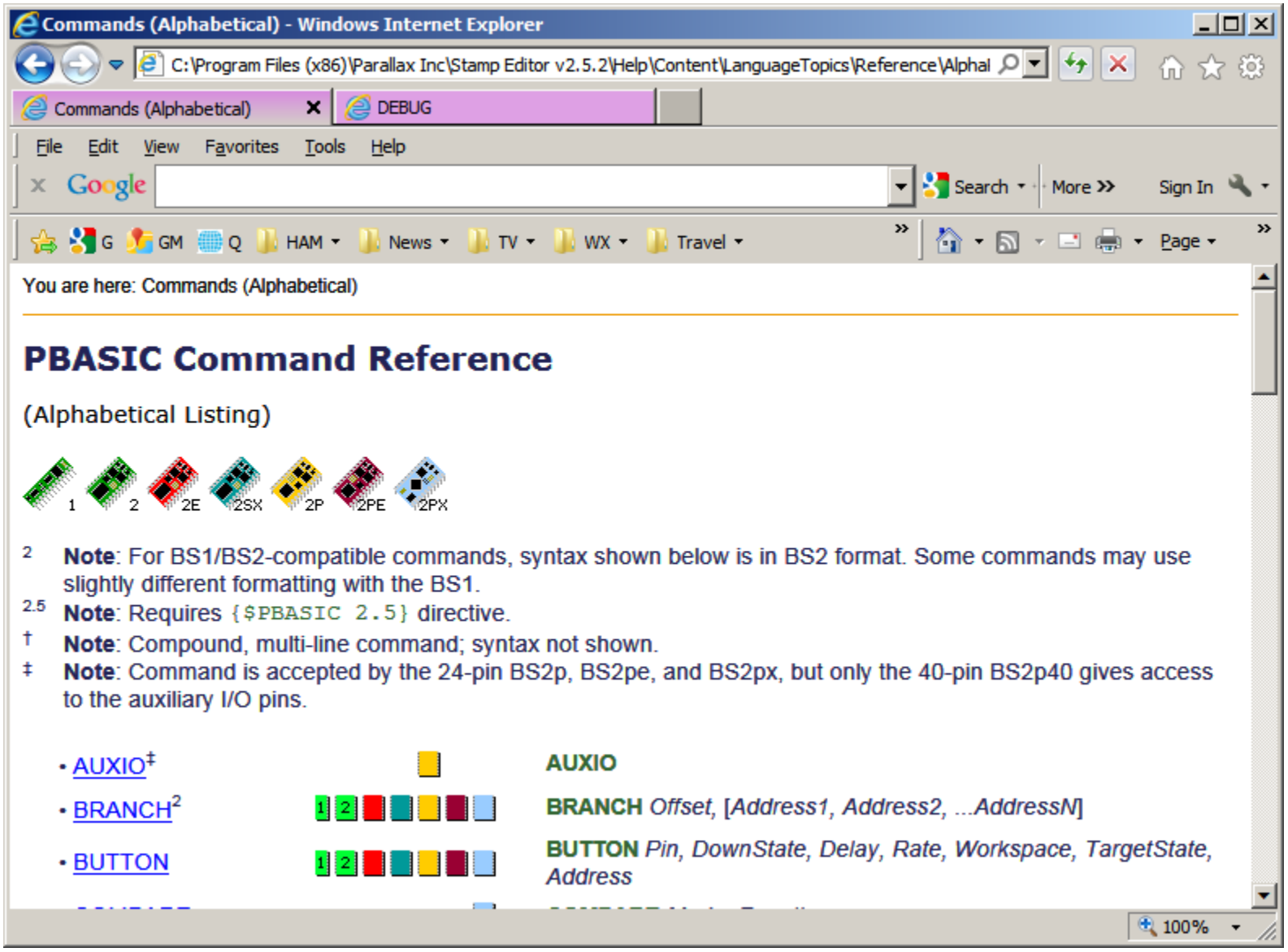
BASIC Stamp Debug Window



Memory MAP


Memory Map - EEPROM 9% Full (LED1.bs2)
EEPROM Map
0 1 2 3 4 5 6 7 8 9 A B C D E F
700
710
720
730 00 00 00 00 00 00 00 00 88 66 45 33 A4 D9 D4 2C
740 69 E6 34 4B 9A 19 CD 7A 04 21 53 CD 6F 20 A4 03
750 46 B2 03 38 C9 2B 33 3F FC 18 A3 99 D1 EC 69 A6
760 8C 68 56 34 63 9A 29 73 9B C9 CD EA 66 54 B3 1E
770 41 C8 54 73 65 E6 85 1F E3 34 7B 9A 29 23 9A 15
780 CD 98 66 CA DC 66 72 B3 BA 19 D5 AC 47 10 32 D5
790 AC 41 7A 60 C2 8F 99 5D B0 6D B4 08 99 6A 06 A6
7A0 C7 FE B2 A9 59 D3 4C 99 DB 2C 6D A6 2C 6F 16 36
7B0 63 9B 95 CD 88 66 3D 82 90 A9 66 92 2E D8 36 5A
7C0 84 4C 35 03 D3 63 7F D9 DC AC 6C D6 36 4B 9B 51
7D0 CD 94 E5 CD DC 66 61 B3 A6 99 B2 BB D9 DB 0C 69
7E0 D6 23 08 99 6A 46 2F 23 9A 15 CD 98 66 CA 8A 66
7F0 4C B3 A3 D9 D1 EC 69 46 35 EB 11 84 4C 35 07 C0
Display ASCII
EEPROM Legend
- Undef. Data
- Def. Data
- Program
- Unused
RAM Map
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
INS:
OUTS:
DIRS:
REG0:
REG1:
REG2:
REG3:
REG4:
REG5:
REG6:
REG7:
REG8:
REG9:
REG10:
REG11:
REG12:
RAM Legend
- Pins
- Nibble
- Word
- Bit
- Byte
- Unused
Source Code
LED1.bs2
Close

IDE Help Window



Debugging – Terminal Input / Output

- Controllability – putting the system into a known state
  - Setting variables
  - Starting program execution
- Observability – knowing what state the system is IN
  - Viewing variables (system state)
  - Viewing current instruction (algorithm state)
  - Viewing inputs, outputs (I/O state)
  - Stopping / pausing program execution
- Tracing – observing the algorithm and system state changes – dynamic behavior
- The DEBUG statement, output format options
- The DEBUGIN statement, input format options
- The PAUSE statement (DEBUGIN can be used also)

PBASIC 2.5 Programming BS2

- Variables: Name VAR Type, Types: Bit (1), Nib (4), Byte (8), Word (16) bits
- Constants: Name CON Constant, Constants: Decimal, %Binary, $Hex, “ASCII”
- Memory: INS, OUTS, DIRS, W0..W12 – 16 words, 32 bytes, 64 nibbles, 256 bits
- Assignment: Variable = Expression
- Output – DEBUG, OUTPUT*
- Input – DEBUGIN, INPUT*
- Statements Unique to BASIC Stamp*
  - Compiler Directives
  - EEPROM and RAM Access*
  - Timing – PAUSE, POLLWAIT
- Linear Sequence of statements
- Change of sequence – (GOTO, GOSUB, RETURN)
- Decision – IF…THEN…ELSE, SELECT…CASE, ON
- Loops – FOR…NEXT, DO…LOOP
- STOP

Expressions

- Data Values
  - Binary
  - HEXadecimal
  - Decimal
  - String
  - ASCII – American Standard Character Information Interchange
  - Constants
  - Variables
  - Arrays
- Operators
  - Binary
  - Unary
  - Precedence
- Algebraic Expressions
- Logical Expression
- String Expressions
- Inequality Expressions

Example Programs

- Hello Program
- Timer
- Digital Output
  - LEDs
  - Speaker / Sound
- Analog Output
- Digital Input
  - Button
  - Switch
- Analog Input
  - Potentiometer
  - Voltage (unable to make example, need more hardware)

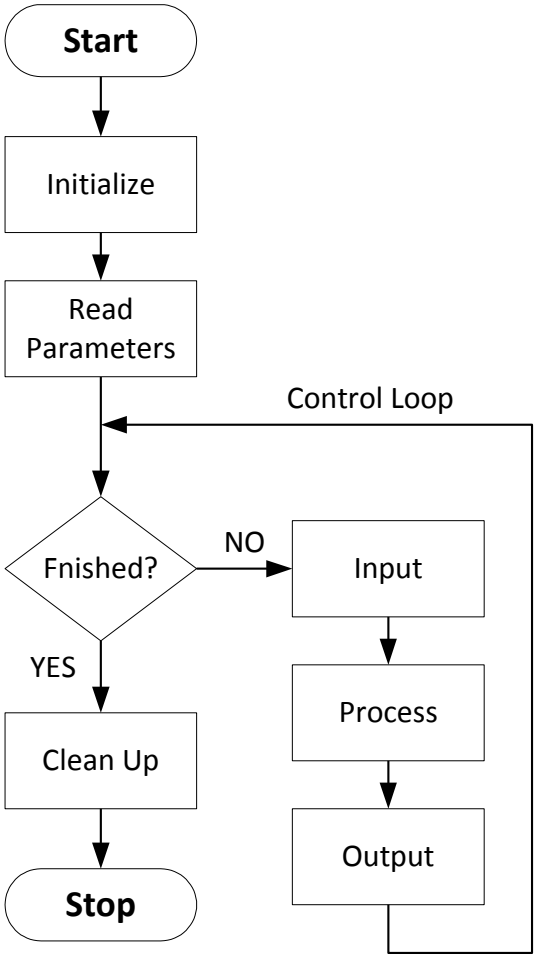
The HELLO Program

- Use the DEBUG Statement to Print to Debug Window
- Become Familiar with:
  - Editor / Explorer
  - Debug Window
  - Memory Map
  - Help System

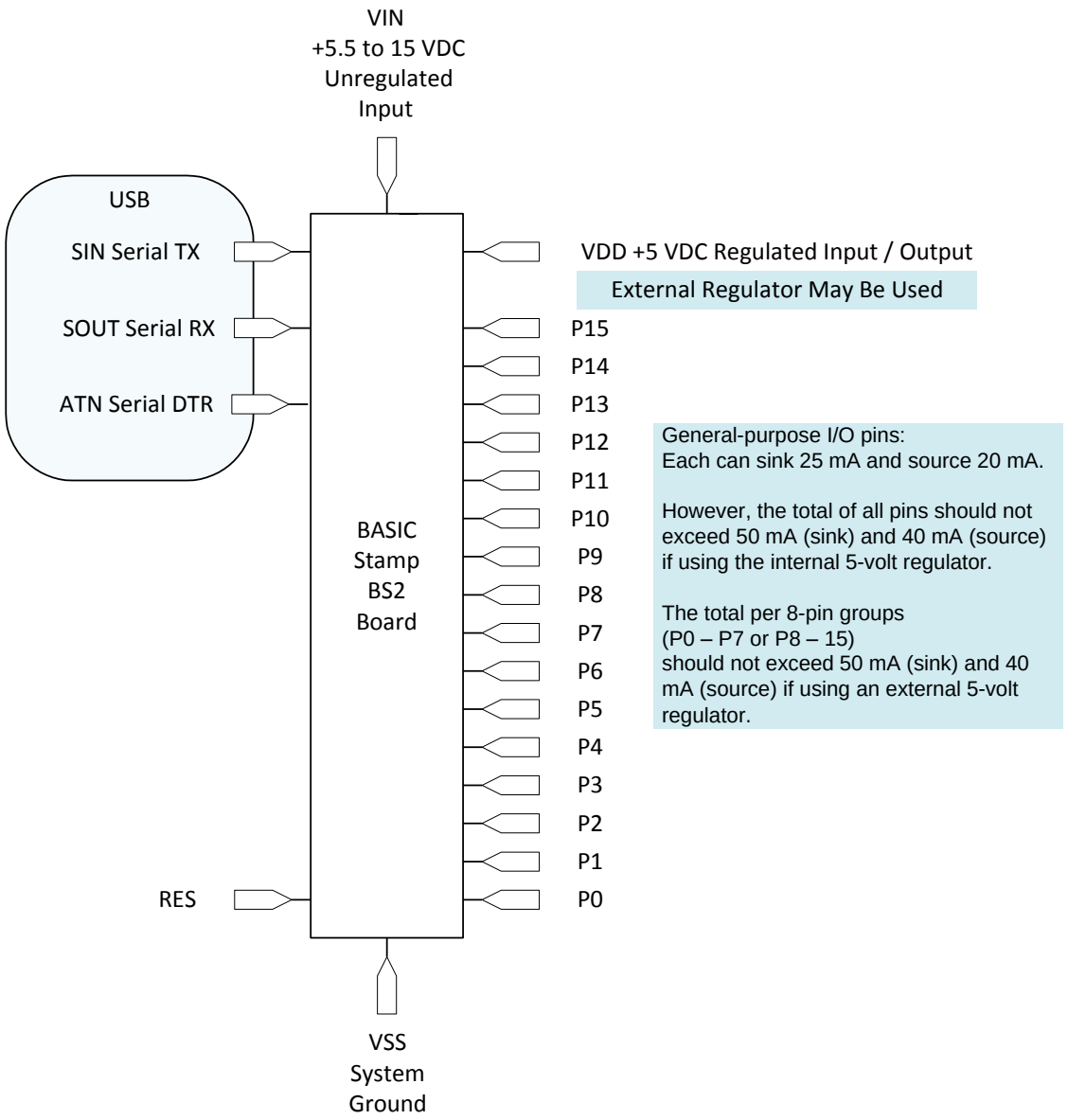
Developing is a Hybrid Activity

- Part Scientist, Engineer, Accountant (rules)
- Part Designer, Artist, Free Thinker (unruly)
- Each Organization Has Its Own Development Standards: Documentation, Naming, Formatting, etc.
- Each Developer Has His Own Style, Make One!
- Be Neat, Orderly, Concise, Consistent, Flexible
- Others May (Have To, Want To) Read Your Work: Document for Maintenance, Reuse
- Each Program is a Work of Art, Literature, Aesthetics
- Don’t Let Rules Get in the Way of the Solution, Creativity

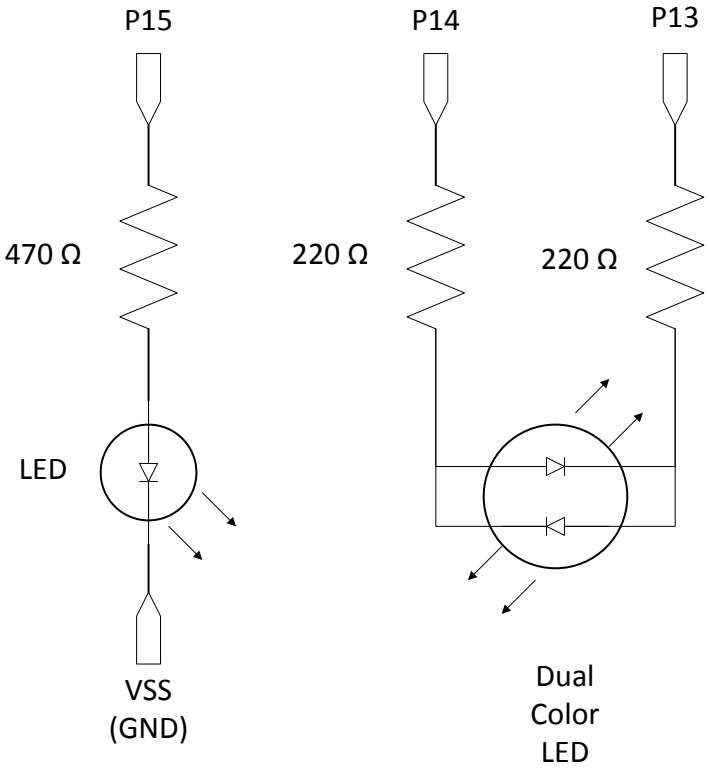
General Overall Program Logic



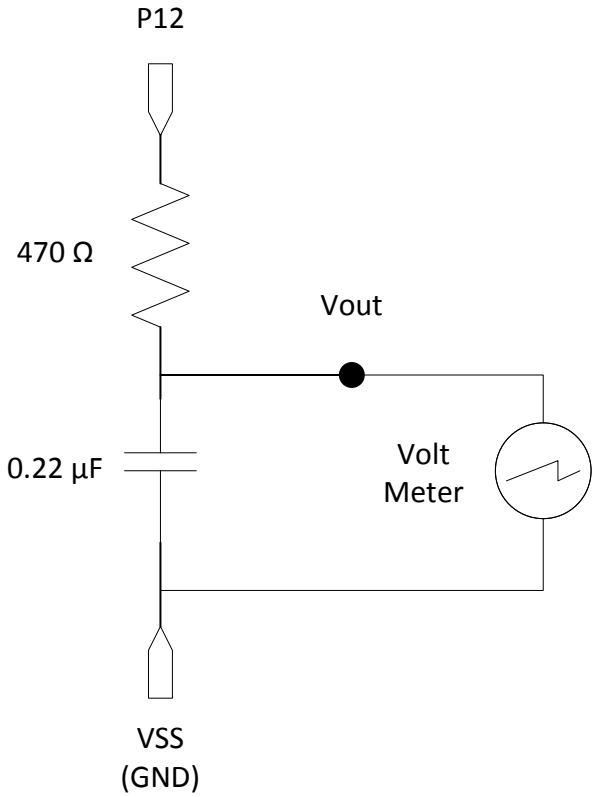
BASIC Stamp BS2 Circuitry

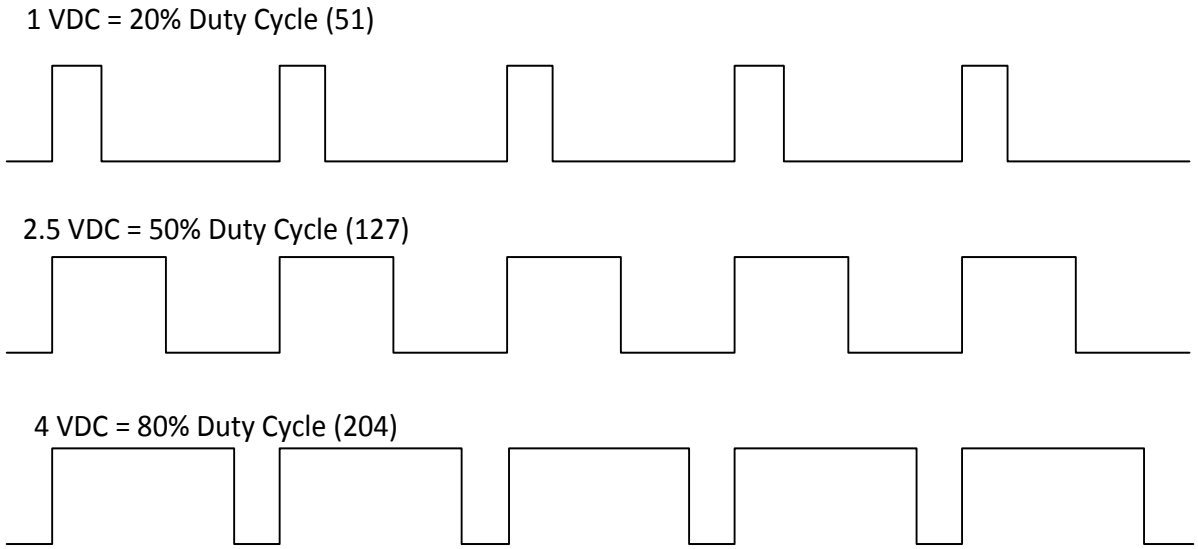


Digital Output to LEDs

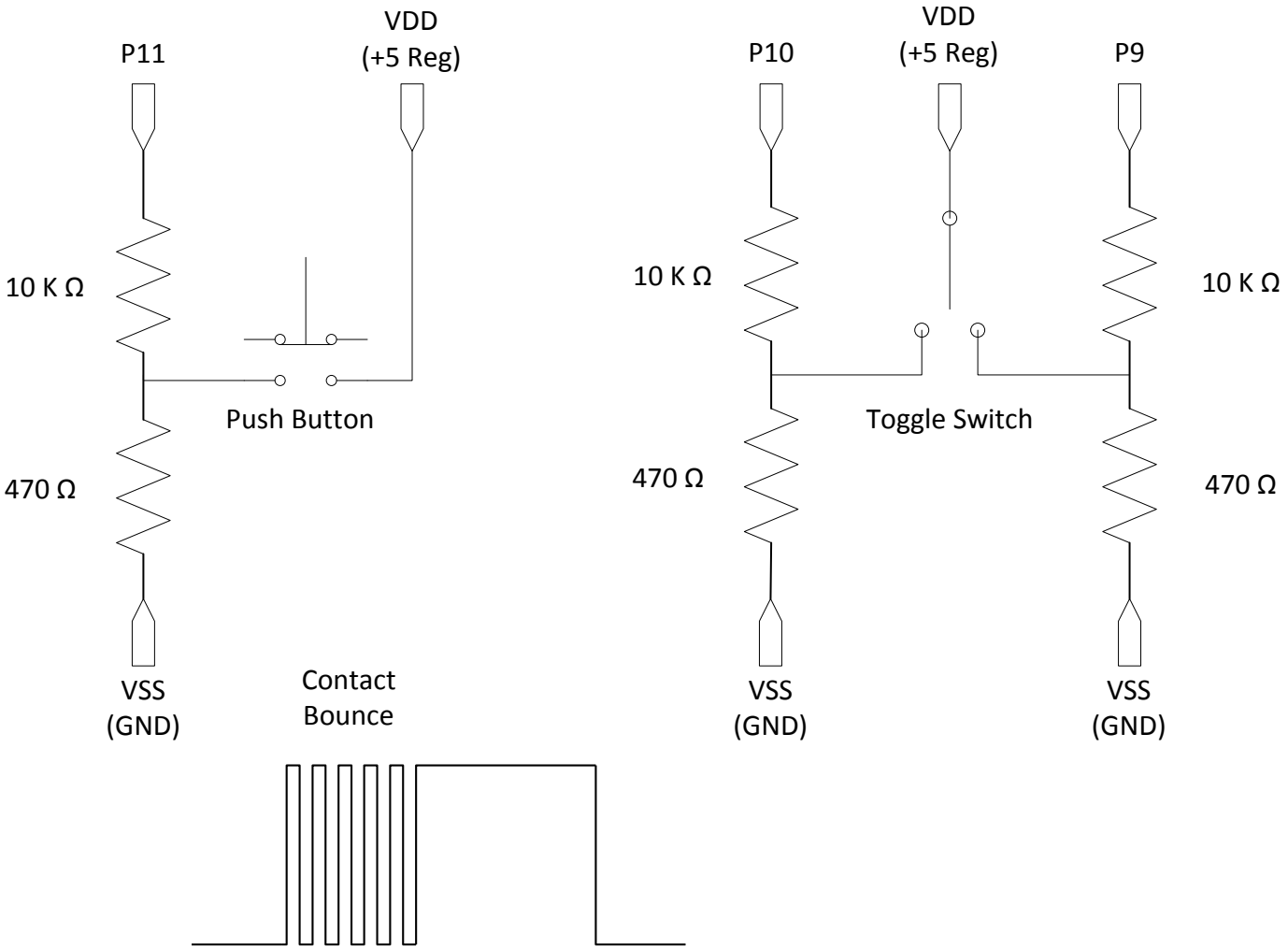


Pulse Width Modulation (PWM) Analog Output

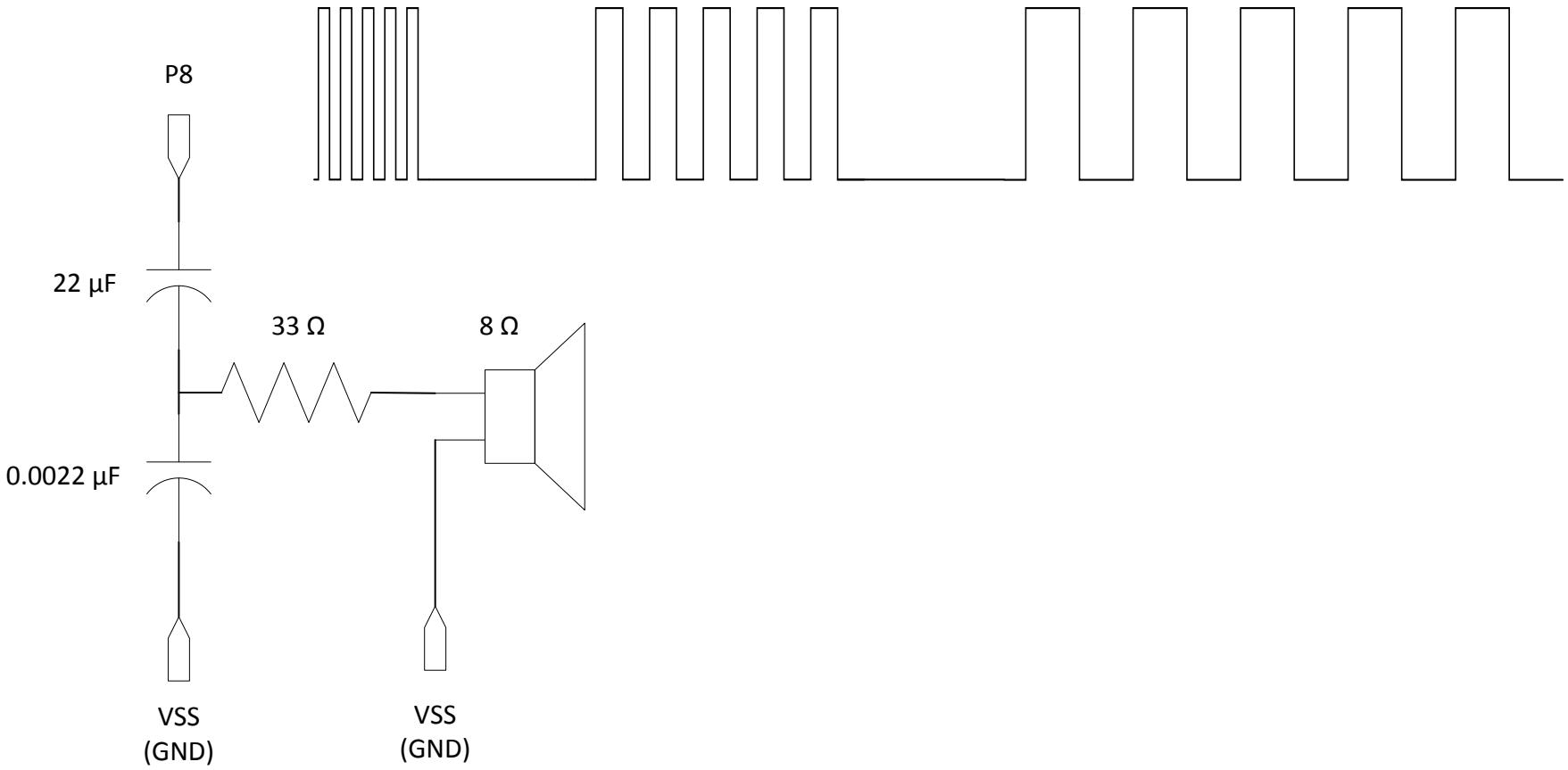




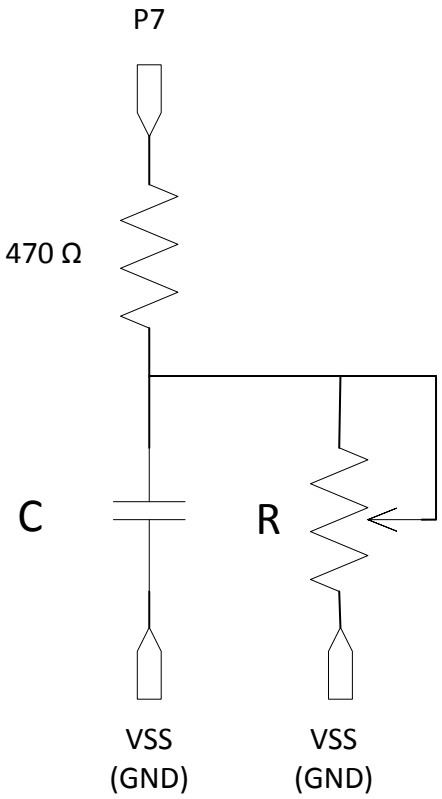
Digital Input From Buttons / Switches



Speaker Output



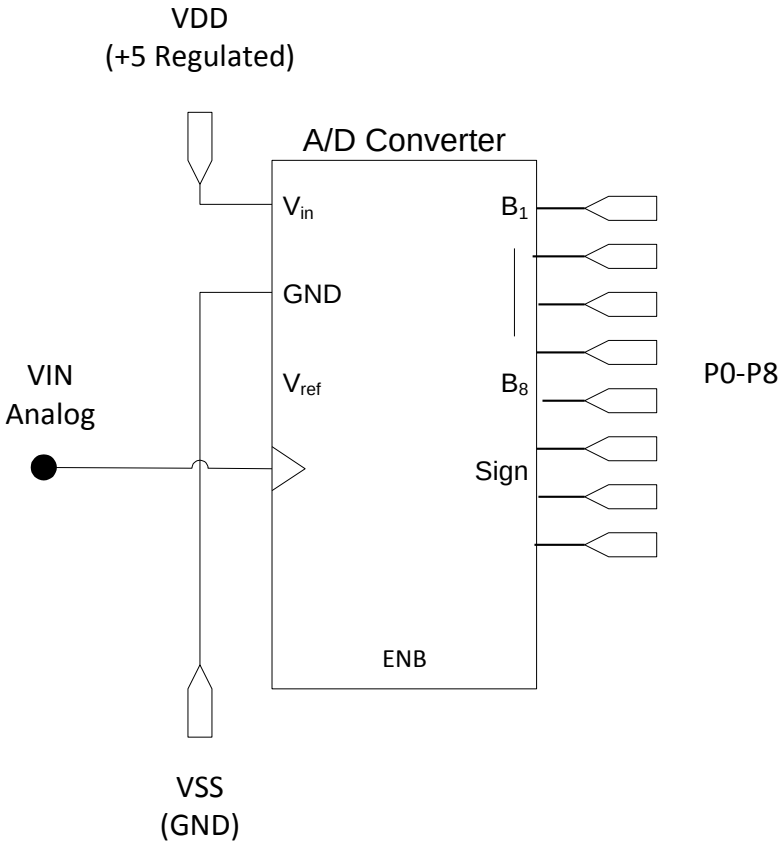
Analog Input – Potentiometer

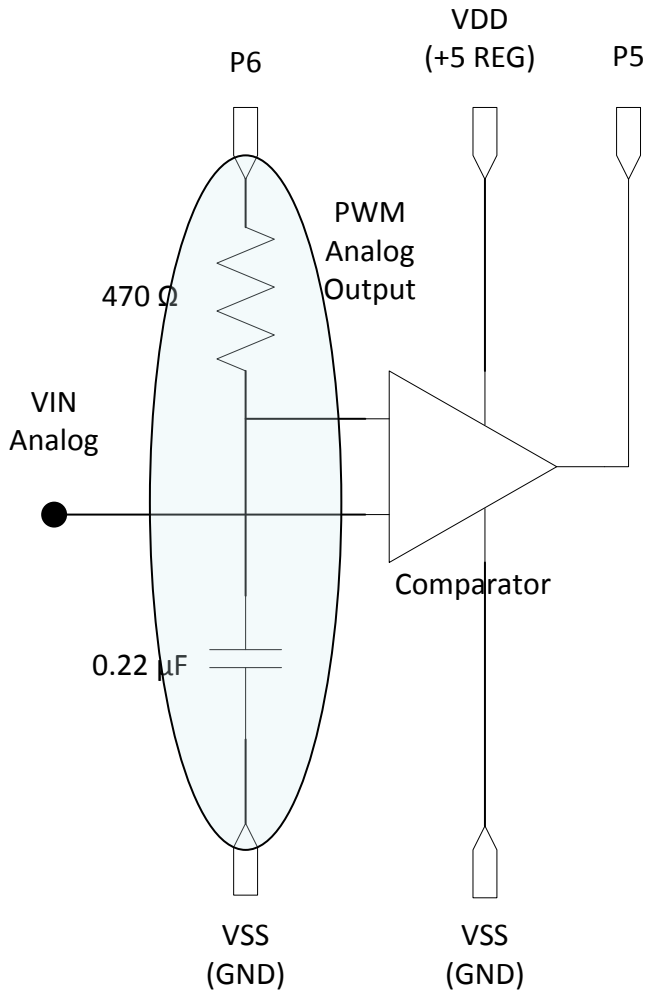


Analog Input – Voltage

Basic Stamp BS2 has NO A/D

Need A/D Chip, Comparator, or OpAmp





Microcontroller Summary (1)

- **Arduino**
  - Available
    - @ Radio Shack, $70 - www.radioshack.com
    - @ Arduino - http://store.arduino.cc/ww/index.php
    - @ Amazon, Digi-key, Mouser, Newark
  - Additional modules needed / available
  - Programmable in C / assembly Language, w/IDE
- **PIC**
  - Available
    - @ ARRL: Kit, $140 – www.arrl.org - Makes CW Keyer
    - @ Microchip - www.microchip.com
    - @ Amazon, Allied Electronics, Avnet, Digi-Key, Mouser, Newark
  - Programmable in C / assembly language, w/IDE (MPLAB)(simulator)
  - PICBASIC language to program PIC is available from meLabs
  - Version of PIC in BASIC Stamp

Microcontroller Summary (2)

- **Parallax Propeller**
  - Available
    - @ Radio Shack, $40 – www.radioshack.com
    - @ Parallax - www.parallax.com
  - Hosts on Windows, Mac, Linux
  - Programmable in SPIN, Assembly Language
  - 32 bit 8 core processor 64KB EEPROM
- **Parallax BASIC Stamp**
  - Available
    - @ Radio Shack, $40 – www.radioshack.com
    - @ Parallax - www.parallax.com
  - Hosts on Windows, Mac, Linux
  - Programmable in PBASIC
  - BS2 Uses PIC16F57 chip, 20MHz Clock, 32Byte RAM, 2K EEPROM
  - Higher level chips use Parallax proprietary SX chips

Question: Can the BASIC Stamp BS2 Be Used for the Automatic Antenna Tuner Project?

- **Consult the Excel Spreadsheet of Antenna Tuner I/O's**
- 3 Motors, 2 Directions Each, 6 out
- Forward Power Analog In, 1 out, 1 in (external comparator)
- Reflected Power Analog In, 1 out, 1 in (external comparator)
- Coil Position Pot Analog In, 1 in/out
- SWR Limit Pot Analog In, 1 in/out
- Capacitor(s) Home Position, 2 in
- AutoTune Start/Stop, 1 in
- **Total 15 I/O – YES, but no band switch (11 in), No Cap/Coil UP/DN (6 in)**
- Is there enough Data Memory? Program Memory? (Dunno?)
- MARGINAL !
- Maybe Use BS2PX which has an analog comparator, need analog switch!
- Maybe Arduino, PIC would be better?