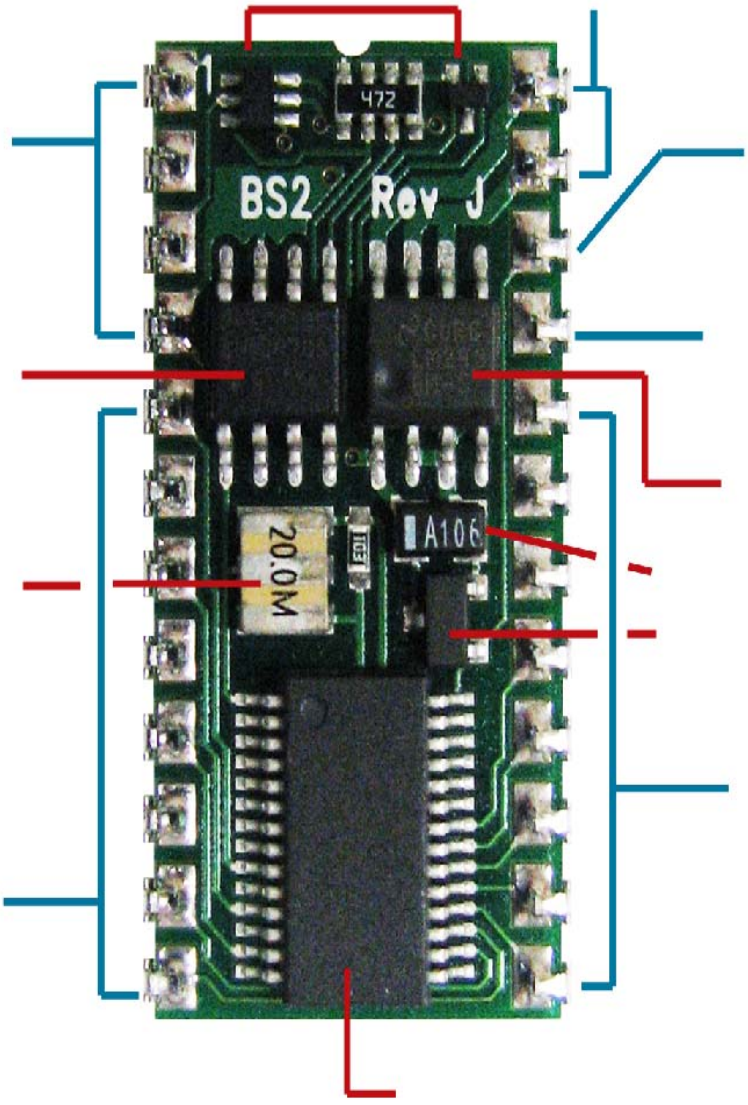
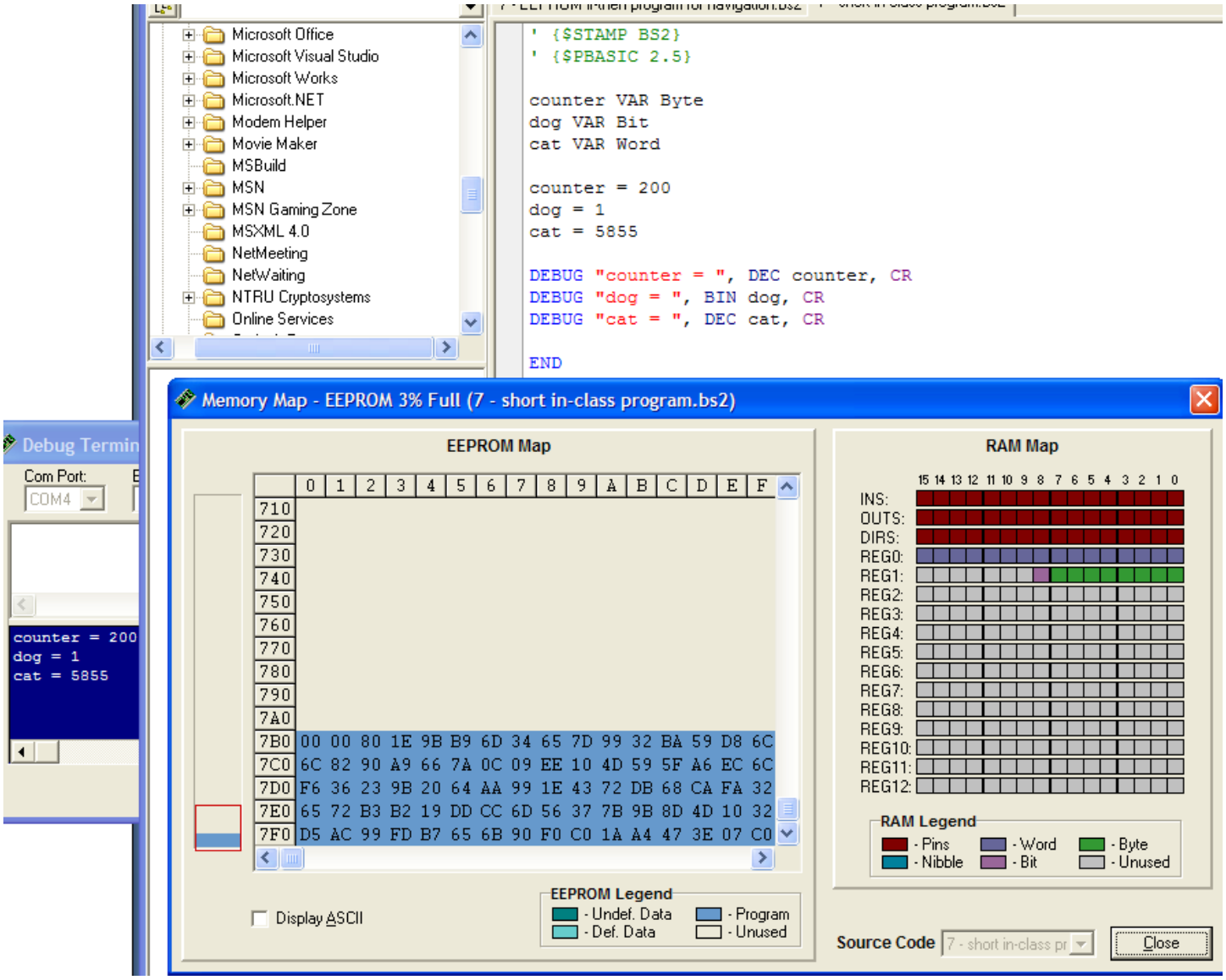
BASIC STAMP MODULE COMPONENTS AND FUNCTION



MEMORY MAP – Enter the program below and look at the memory map. Put in one variable at a time and see what happens.



EEPROM = Electrically Erasable Read Only Memory

- can hold of information
- primary function is to store
- can also store (storage starts at the top and works down) using DATA command
- read data from EEPROM using command
- won't be erased when you the Boe-Bot
- slow compared to
- breaks after about 10 million writes (RAM is unlimited)

RAM = Access Memory

- of RAM
- bits is reserved (will discuss later)
- bits is actually usable

Learn about how the program below works, and examine the memory map while the program is running.

EEPROM Navigation Program

```
' Robotics with the Boe-Bot - EepromNavigation.bs2
' Navigate using characters stored in EEPROM.

' {$STAMP BS2}                              ' Stamp directive.
' {$PBASIC 2.5}                             ' PBASIC directive.

' -----[ Variables ]-------------------------------------------------------

pulseCount      VAR     Word                ' Stores number of pulses.
address         VAR     Byte                ' Stores EEPROM address.
instruction     VAR     Byte                ' Stores EEPROM instruction.

' -----[ EEPROM Data ]-----------------------------------------------------

'      Address: 0123456789                 ' These two commented lines show
'               ||||||||||                 ' EEPROM address of each datum.
DATA            "FLFFRBLBBQ"                ' Navigation instructions.

' -----[ Initialization ]---------------------------------------------------

FREQOUT 4, 2000, 3000                       ' Signal program start/reset.

' -----[ Main Routine ]-----------------------------------------------------

DO

  READ address, instruction                 ' Data at address in instruction.
  address = address + 1                     ' Add 1 to address for next read.

  SELECT instruction
    CASE "F": GOSUB Forward
    CASE "B": GOSUB Backward
    CASE "L": GOSUB Left_Turn
    CASE "R": GOSUB Right_Turn
  ENDSELECT

LOOP UNTIL instruction = "Q"

END                                         ' Stop executing until reset.


' -----[ Subroutine - Forward ]--------------------------------------------

Forward:                                    ' Forward subroutine.
  FOR pulseCount = 1 TO 64                  ' Send 64 forward pulses.
    PULSOUT 13, 850                         ' 1.7 ms pulse to left servo.
    PULSOUT 12, 650                         ' 1.3 ms pulse to right servo.
    PAUSE 20                                ' Pause for 20 ms.
  NEXT
  RETURN                                    ' Return to Main Routine loop.

' -----[ Subroutine - Backward ]-------------------------------------------

Backward:               ' Backward subroutine.
  FOR pulseCount = 1 TO 64                  ' Send 64 backward pulses.
    PULSOUT 13, 650                         ' 1.3 ms pulse to left servo.
    PULSOUT 12, 850                         ' 1.7 ms pulse to right servo.
    PAUSE 20                                ' Pause for 20 ms.
  NEXT
  RETURN                                    ' Return to Main Routine loop.

' -----[ Subroutine - Left turn ]------------------------------------------

Left_Turn:                                  ' Left turn subroutine.
  FOR pulseCount = 1 TO 24                  ' Send 24 left rotate pulses.
    PULSOUT 13, 650                         ' 1.3 ms pulse to left servo.
    PULSOUT 12, 650                         ' 1.3 ms pulse to right servo.
    PAUSE 20                                ' Pause for 20 ms.
  NEXT
  RETURN                                    ' Return to Main Routine loop.

' -----[ Subroutine - Right turn ]-----------------------------------------

Right_Turn:                                 ' right turn subroutine.
  FOR pulseCount = 1 TO 24                  ' Send 24 right rotate pulses.
    PULSOUT 13, 850                         ' 1.7 ms pulse to left servo.
    PULSOUT 12, 850                         ' 1.7 ms pulse to right servo.
    PAUSE 20                                ' Pause for 20 ms.
  NEXT
  RETURN                                    ' Return to Main Routine section.
```

Making Plots Using Excel

CLASS EXAMPLE:

Recall the problem from the last class where you computed the velocity of a falling body as

$$V = V_0 + g \cdot t$$

where V_0 = 0 m/s (initial velocity)

g = 9.81 m/s^2 (gravitational constant)

t = 0 to 15 s (elapsed time of flight)

Using Excel, we created the following table . . .

Velocity (m/s)	Time (s)	Initial Velocity (m/s)	Gravity (m/s^2)
0.00	0	0	9.81
9.81	1		
19.62	2		
29.43	3		
39.24	4		
49.05	5		
58.86	6		
25.00	7		
78.48	8		
88.29	9		
98.10	10		
107.91	11		
117.72	12		
127.53	13		
137.34	14		
147.15	15		

Today, we would like to use Microsoft Excel worksheet to plot the velocity of the falling body versus the time of flight.

Rework the Skydiving Problem Using Mathcad

Keystrokes

$V_0 := 0 \cdot \frac{m}{s}$ V.0:0*m/s <enter>

$V(t) := V_0 + g \cdot t \cdot s$ V(t):V.0+g*t*s <enter>

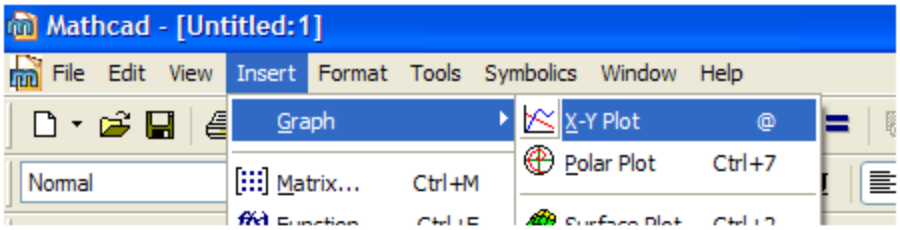
TIPS:

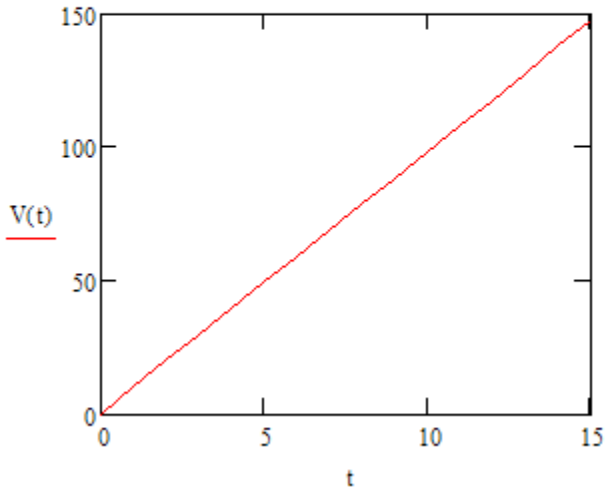
g is a predefined constant (9.81 m/s2)

Make sure you put the unit of seconds on the time t

$t := 0, 1 .. 15$ t:0,1;15 <enter>

Insert the Plot:





Enter t and V(t) in the placeholders that are provided.

GROUP PROBLEM:

An engineer graduated in May 2007 and went to work for a company making about $55,000 a year. Suppose this engineer works really hard and earns a raise of 5% for EACH year that he works! Plot this engineer's annual salary for the first 25 years of his career.

HINT: You must make a table first. The first year the graduate makes $55,000, the second year the graduate makes 1.05 x $55,000, the third year the graduate makes 1.05 x 1.05 x $55,000, and so on.

NOTE: Your actual earnings may not match those in this academic scenario.

After you finish, try to repeat this problem using Mathcad.