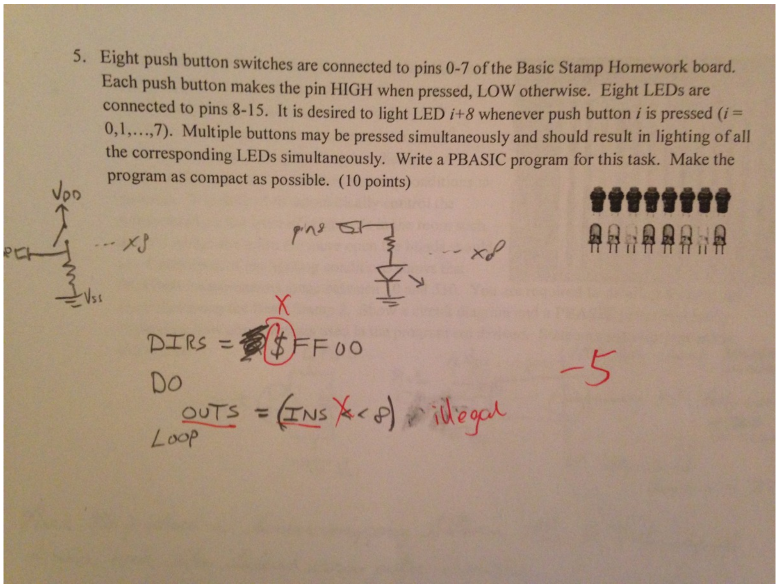
Ben Lorenzetti
Embedded Systems Lab

Exam 1 Regrading

On exam one, I think that 7 of the points taken off were not justified.

Problem 5: 5pts off



5. Eight push button switches are connected to pins 0-7 of the Basic Stamp Homework board. Each push button makes the pin HIGH when pressed, LOW otherwise. Eight LEDs are connected to pins 8-15. It is desired to light LED *i+8* whenever push button *i* is pressed (*i* = 0,1,…,7). Multiple buttons may be pressed simultaneously and should result in lighting of all the corresponding LEDs simultaneously. Write a PBASIC program for this task. Make the program as compact as possible. (10 points)

```
DIRS = $FF00
DO
  OUTS = (INS <<8)
LOOP
```

X

illegal

−5

The dollar sign is the hexadecimal indicator in PBASIC. OUTS and INS are both valid, Word-size registers and bitshift left is a valid operation in PBASIC, used just like it is in C.

To demonstrate that my code works, I loaded it onto my Basic STAMP board and recorded a video. See attached file: **Problem-5-Demo.MOV.**

Problem 6: 2pts off

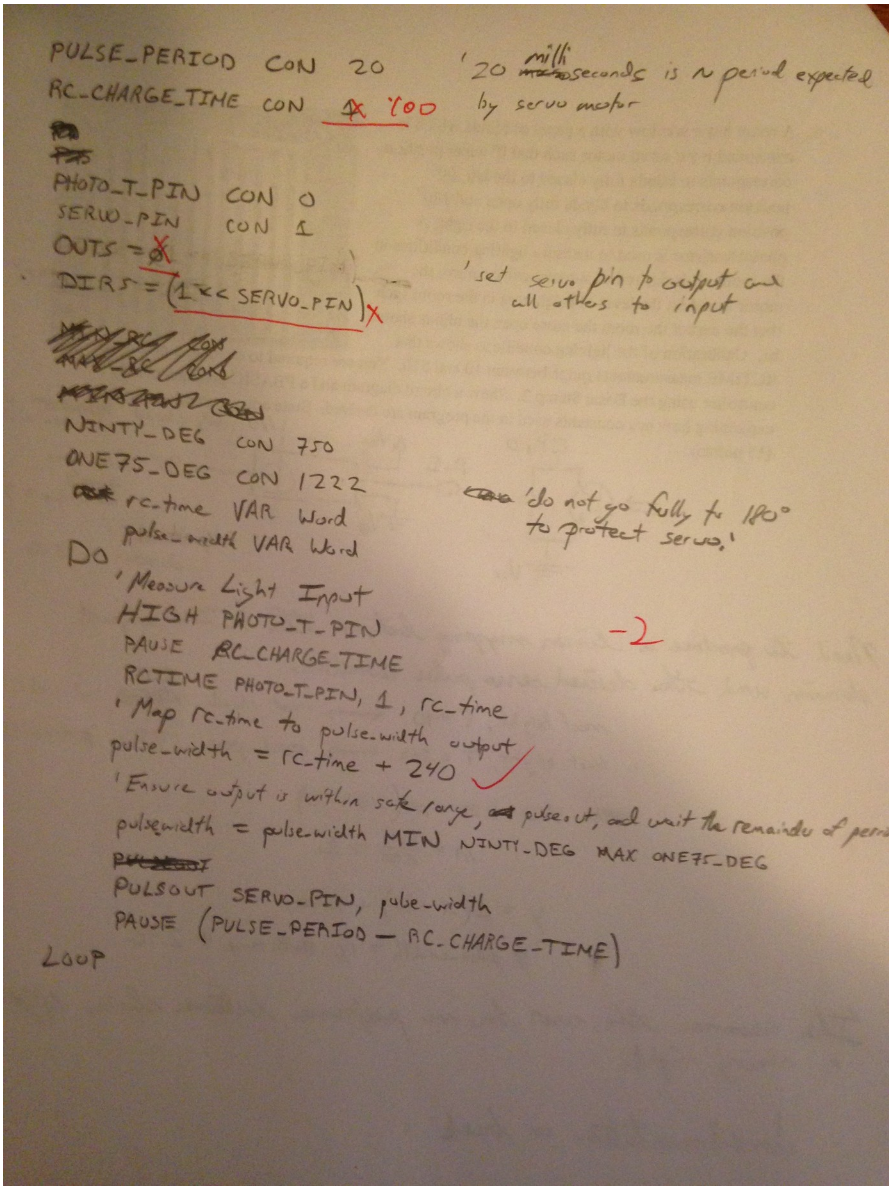
I lost 2 points on my answer for problem 6.

The first point was apparently for having too small a charge time. I put 1 millisecond and the grader wrote 100. However, for lab 3 I used 1 millisecond and it worked just fine. You can see my lab 3 implementation (T_RC_CHARGE CON 1) on page 6 of the attached file **lab-3-report.pdf**. Assuming a current limiting resistor of 220 ohms,

$$5\tau = 5*RC = 5 * 0.1uF * 220\ \Omega = 0.11\ ms$$

So 1 ms is more than adequate.

The second point I missed was for essentially the same code I was marked for in problem 5.



```
PULSE_PERIOD CON 20          '20 ~~micro~~milliseconds is ~ period expected
RC_CHARGE_TIME CON ~~1~~ 100      by servo motor

PHOTO_T_PIN CON 0
SERWO_PIN CON 1
OUTS = ~~0~~
DIRS = (1 << SERVO_PIN)  X         'set servo pin to output and
                                     all others to input

NINTY_DEG CON 750
ONE75_DEG CON 1222                 'do not go fully to 180°
  rc_time VAR Word                   to protect servo.'
  pulse_width VAR Word
Do
  'Measure Light Input
  HIGH PHOTO_T_PIN                           -2
  PAUSE RC_CHARGE_TIME
  RCTIME PHOTO_T_PIN, 1, rc_time
  'Map rc_time to pulse_width output
  pulse_width = rc_time + 240  ✓
  'Ensure output is within safe range, pulseout, and wait the remainder of peri
  pulsewidth = pulse_width MIN NINTY_DEG MAX ONE75_DEG
  PULSOUT SERVO_PIN, pulse_width
  PAUSE (PULSE_PERIOD - RC_CHARGE_TIME)
LOOP
```