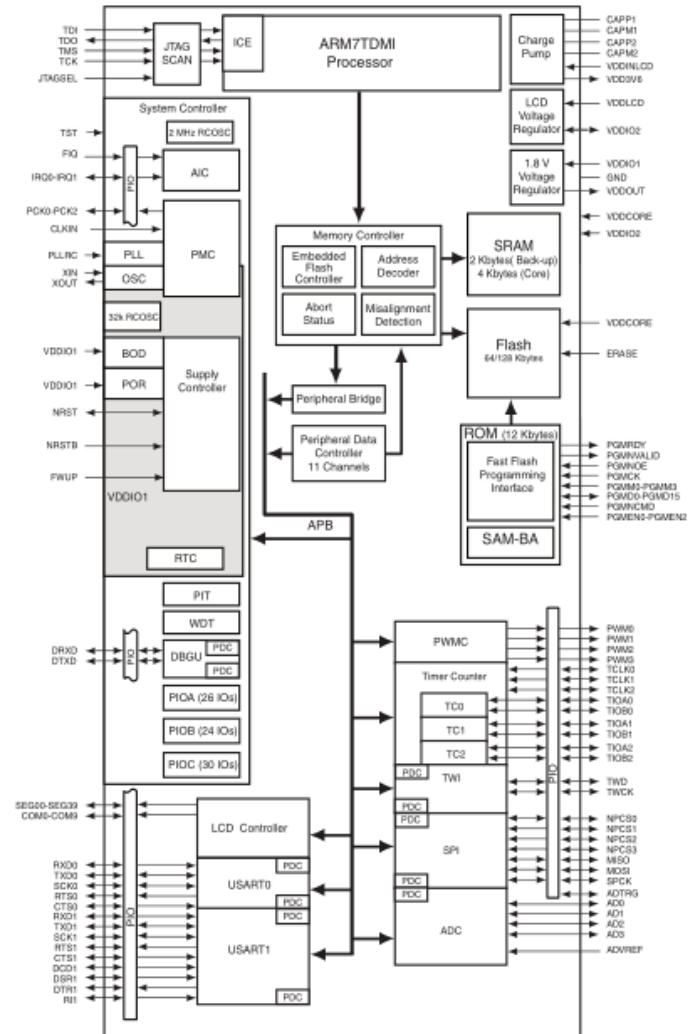
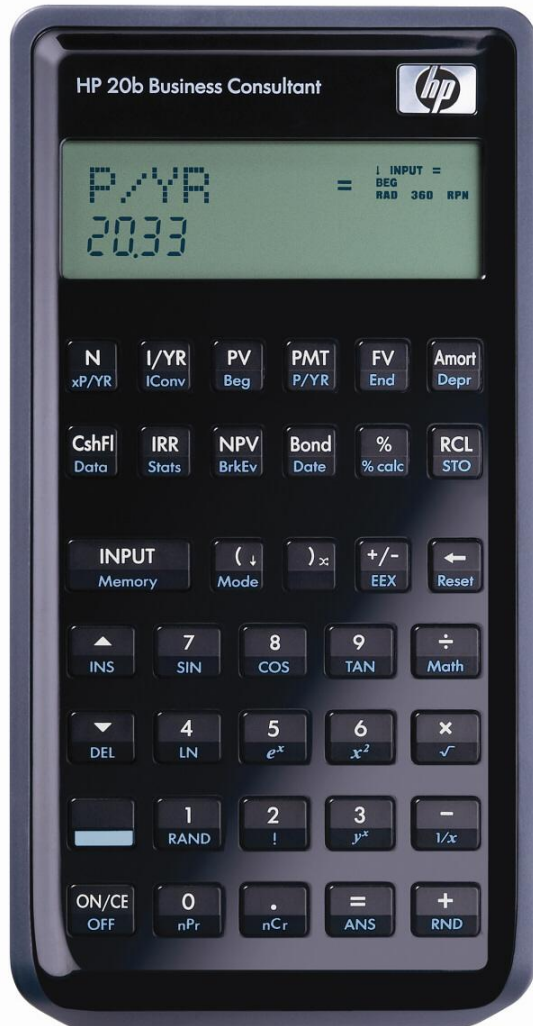


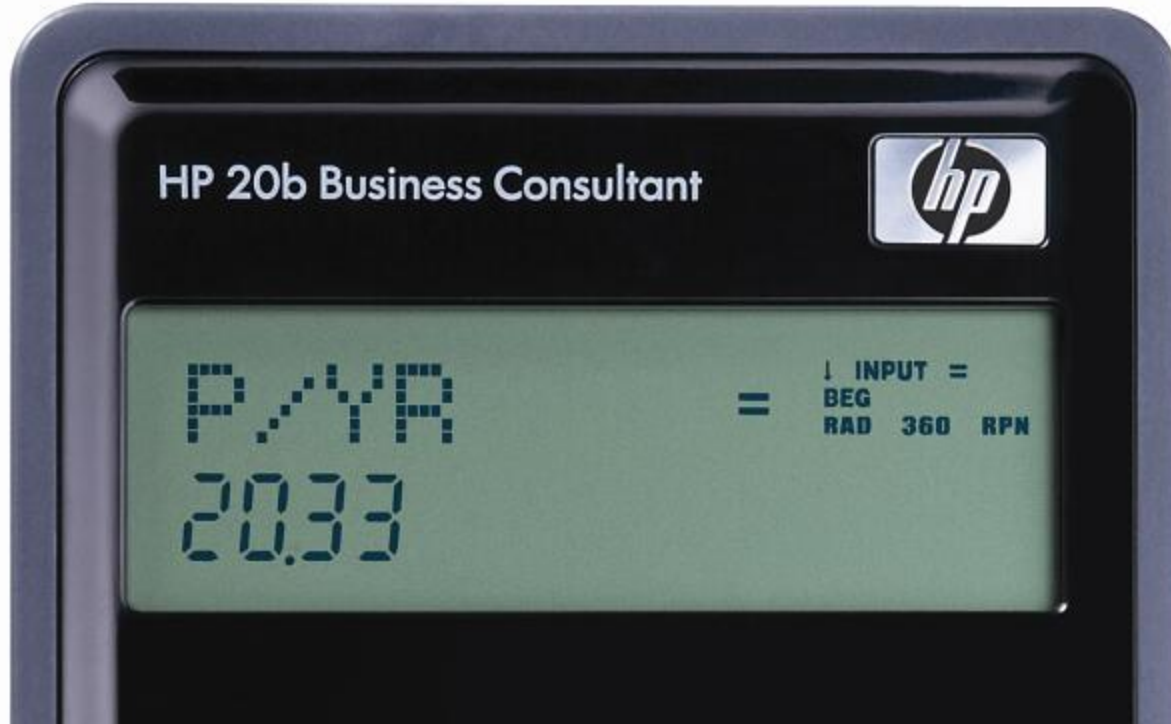
# ENGI E1112 Departmental Project: Computer Science/Computer Engineering

By Kaiven Zhou, Alex Ge, Anna Teng

# Introduction



# THE LCD DISPLAY



1. `void lcd_init()`

2. `void lcd_put_char7(char ch, int col)`

3. `void lcd_print7(const char *c)`

# Lab 1: A Scrolling Display

//By: Abhinav Mishra, Andrew Pope, Yiming Ge, Anna Teng, Will VanArsdall, Kaiven Zhou

```
int strlen(const char *s)
{
    int n;
    for(n=0; *s!='\0';s++)
        n++;
    return n;
}

int main()
{
    lcd_init();

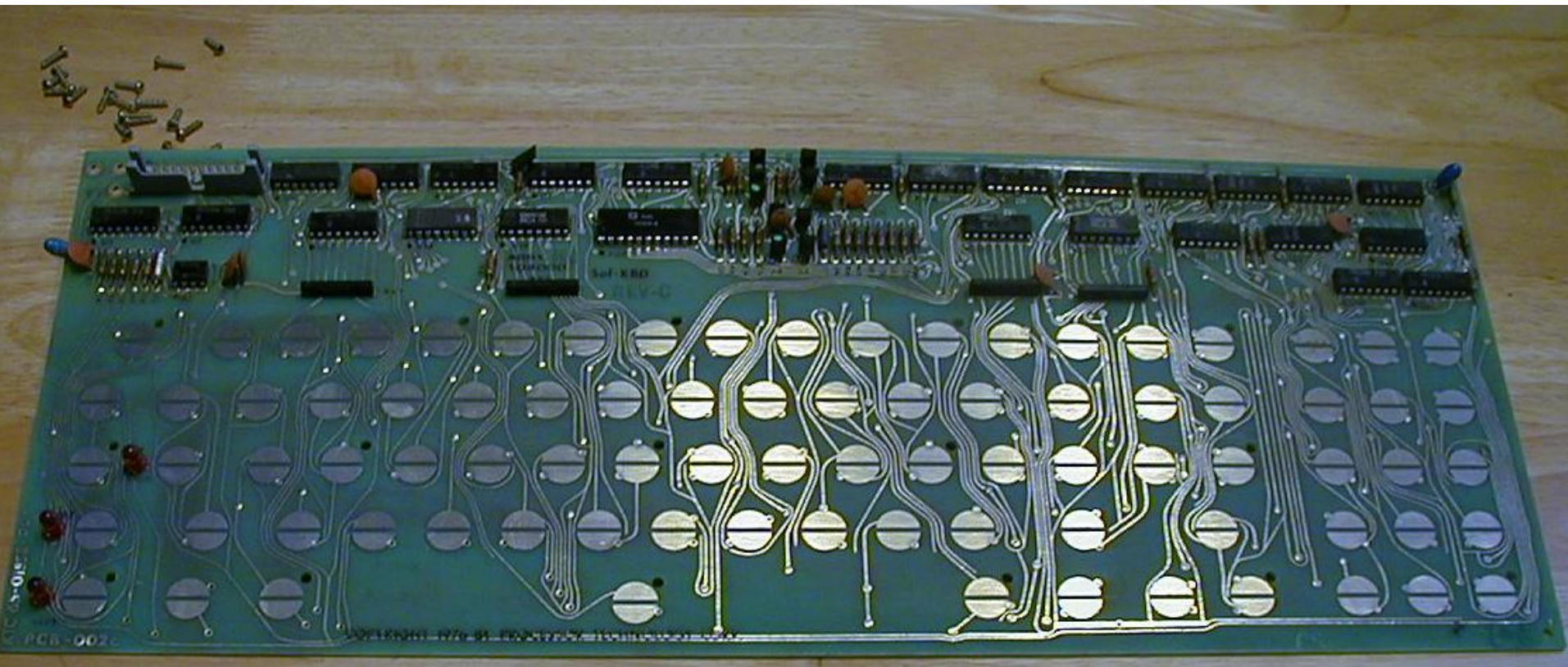
    char test[] = "test";           //the message
    int len = 0;
    len = strlen(test);           //length of the message
    int x=0;                       //the starting position of the first character
```

```
while(1)
{
    int i, t, j;
    t=x;

    for(i=0; i < len; i++)
    {
        lcd_put_char7(test[i],t%12);
        t++;
    }

    while(j<50000)    j+=1;
    lcd_print7("    "); //clears the screen
    j=0;
    x++;
}
return 0;
}
```

# Lab 2: Scanning the Keyboard



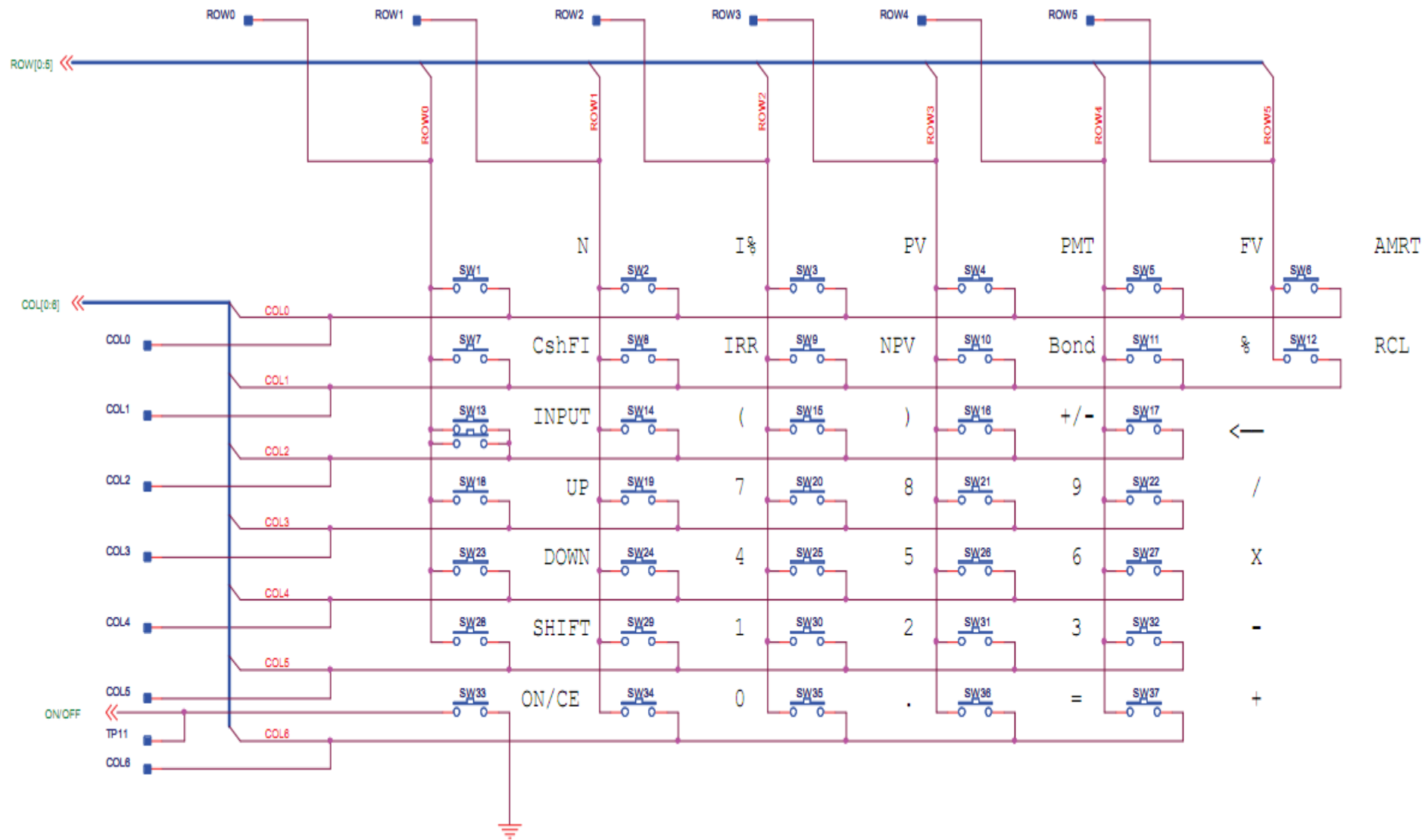
“rows”

**PC11**    **PC12**    **PC13**    **PC14**    **PC15**    **PC26**

“columns”

|            |       |      |     |      |    |       |
|------------|-------|------|-----|------|----|-------|
| <b>PC0</b> | N     | I/YR | PV  | PMT  | FG | Amort |
| <b>PC1</b> | CshFl | IRR  | NPV | Bond | %  | RCL   |
| <b>PC2</b> | INPUT | (    | )   | +/-  | ←  |       |
| <b>PC3</b> | ▲     | 7    | 8   | 9    | ÷  |       |
| <b>PC4</b> | ▼     | 4    | 5   | 6    | ×  |       |
| <b>PC5</b> | shift | 1    | 2   | 3    | -  |       |
| <b>PC6</b> |       | 0    | .   | =    | +  |       |





- `Keyboard_init()`
- `Keyboard_column_low(int)`
- `Keyboard_column_high(int)`
- `Keyboard_row_read(int)`

```
#define NUM_COLUMNS 7
#define NUM_ROWS 6
const char* keys[44] = {"", "N", "I/YR", "PV", "PMT", "FG",
    "Amort",
    "CshFl", "IRR", "NPV", "Bond", "%", "RCL",
    "INPUT", "(", ")", "+/-", "<-", "",
    "UP", "7", "8", "9", "/", "",
    "DOWN", "4", "5", "6", "x", "",
    "SHIFT", "1", "2", "3", "-", "",
    "", "0", ".", "=", "+", ""};
```

```
int keyboard_key()
{
    int i, j;
    int x = 1;

    for(i = 0; i < NUM_COLUMNS; i++)
    {
        keyboard_column_low(i);

        for(j = 0; j < NUM_ROWS; j++)
        {
            if(!keyboard_row_read(j))
            {
                keyboard_column_high(i); //Resets the current column
                return x;
            }
            x++;
        }

        keyboard_column_high(i);
    }
    return 0;
}
```

# Lab 3: Entering and Displaying Numbers

- Was a number entered?
- Was the change sign key pressed?
- Was an operation entered?
- Was a garbage key pressed?

```
//By Kaiven Zhou, Yiming Ge, Anna Teng
```

```
void keyboard_get_entry(struct entry *result)
```

```
{
```

```
    int inputNumber=INT_MAX;
```

```
    int lastKey=-1,inputOperation=-1,numOfDigits=0;
```

```
    lcd_put_char7(0,'+'); //by default the number is positive
```

```
while(1)
```

```
{
```

```
    while(keyboard_key()); //ensure no key is pressed
```

```
    while(lastKey===-1) lastKey = keyboard_key(); //get key
```

```
if('0'<=lastKey && lastKey <= '9' && numOfDigits<=MAX_NUM_DIGITS)
{
    if(inputNumber==INT_MAX)
        inputNumber=0;

    int integerOfLastKey = lastKey-'0'; //char to int

    inputNumber*=10;
    inputNumber+=integerOfLastKey;

    lcd_put_char7(lastKey,numOfDigits+1); //the newest
    digit placed should go after the sign and the previous number (which
    has numOfDigits digits)

    numOfDigits++;
}
```

```
else if(lastKey == '~')
    {
        if(inputNumber==INT_MAX)
            inputNumber=0;

        inputNumber*=-1;

        if(inputNumber<0)
            lcd_put_char7(0,'-');
        else
            lcd_put_char7(0,'+');
    }
else if(lastKey=='+' || lastKey=='-' || lastKey=='*' || lastKey=='/' || lastKey=='\r')
    {
        result->number = inputNumber;
        result->operation = inputOperation;
        return;
    }
```



# Lab 4: An RPN Calculator

What is Reverse Polish Notation (RPN)?

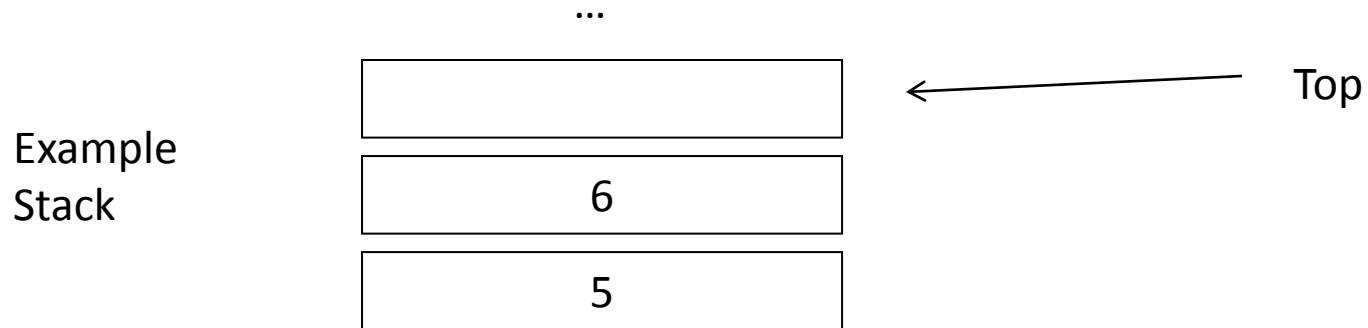
Normally we would enter (infix notation):

$1 + 2 - 3$

In Reverse Polish Notation:

$1 \text{ return } 2 + 3 -$

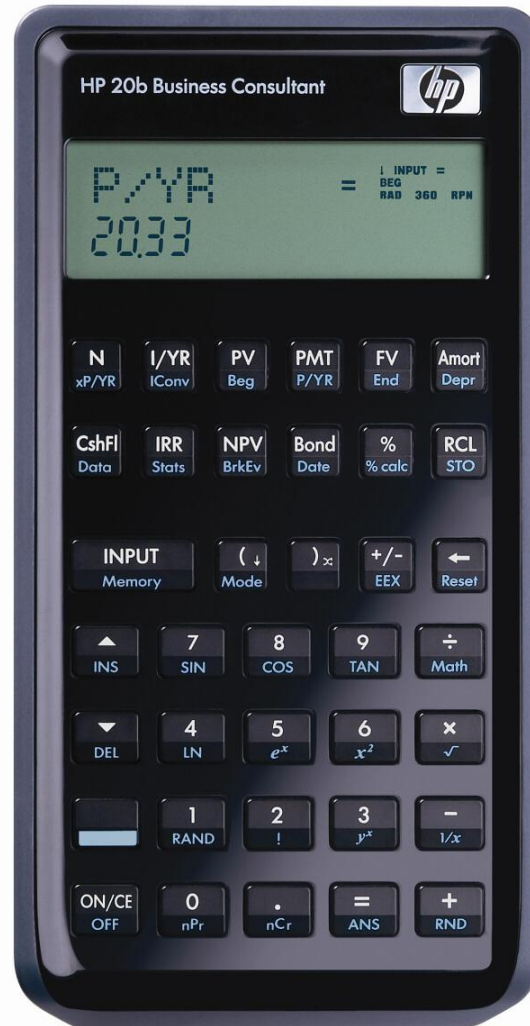
# Lab 4: An RPN Calculator



```
void rpn(int *stack, int *top, struct entry *input)
{
    if(input->number != INT_MAX) //if there is a number, add it to the stack
    {
        stack[(*top)] = input->number;
        (*top)++;
    }
}
```

```
if( (*top)>1 ) //only do an operation if there are at least 2 numbers in the stack
{
    switch(input->operation)
    {
        case '\r': break;           //done in the first if statement
        case '+': lcd_print_int( stack[( *top)-2] += stack[( *top)-1] );
                    (*top)--;
                    break;
        case '-': lcd_print_int( stack[( *top)-2] -= stack[( *top)-1] );
                    (*top)--;
                    break;
        case '/': lcd_print_int( stack[( *top)-2] /= stack[( *top)-1] );
                    (*top)--;
                    break;
        case '*': lcd_print_int( stack[( *top)-2] *= stack[( *top)-1] );
                    (*top)--;
                    break;
    }
}
```

# Our RPN Calculator



Lessons  
Learned