**Redesign of IOT wave probe signal conditions**

Collett, C.



**Publisher's version / Version de l'éditeur:**
https://doi.org/10.4224/8895623
*Student Report (National Research Council of Canada. Institute for Ocean Technology); no. SR-2005-04, 2005*

# DOCUMENTATION PAGE

| REPORT NUMBER | NRC REPORT NUMBER | DATE | | |
|---|---|---|---|---|
| SR-2005-04 | SR-2005-04 | April 21, 2005 | | |
| **REPORT SECURITY CLASSIFICATION** | | **DISTRIBUTION** | | |
| Unclassified | | Unlimited | | |
| **TITLE** | | | | |
| Design and Testing of Alternate Wave Probe Signal Conditioner | | | | |
| **AUTHOR(S)** | | | | |
| Chad Collett | | | | |
| **CORPORATE AUTHOR(S)/PERFORMING AGENCY(S)** | | | | |
| **PUBLICATION** | | | | |
| **SPONSORING AGENCY(S)** | | | | |
| **IOT PROJECT NUMBER** | | **NRC FILE NUMBER** | | |
| 421008 | | | | |
| **KEY WORDS** | | **PAGES** | **FIGS.** | **TABLES** |
| Wave, Probe, Conditioner, PIC, RS232 | | 43 | 11 | 9 |


**SUMMARY**

An accurate and reliable way of measuring wave properties is fundamental for proper evaluation of ship hull designs and tank tests where specific types of waves are generated. The repeatability of wave probe data is essential for evaluating test results. These important qualities are in question for the current systems.

In this report I tested the effects of using thicker gauge wire on the IOT wave probes. The results are outlined within the body of the report and some conclusions are made from them. I also designed, prototyped and tested a new IOT wave probe signal conditioner. Also there were a couple of new ideas, which I developed that can be used for other projects and circuits. All of which are covered within this report.



**ADDRESS**
National Research Council
Institute for Ocean Technology
Arctic Avenue, P. O. Box 12093
St. John's, NL A1B 3T5
Tel.: (709) 772-5185, Fax: (709) 772-2462

National Research Council Canada
Conseil national de recherches Canada

Institute for Ocean Technology
Institut des technologies océaniques

# Design and Testing of Alternate Wave Probe Signal Conditioner

**SR-2005-04**

Chad Collett

April 2005

## Acknowledgements

In no particular order:
Greg Janes
Peter Hacket
Tim Ennis
Jim Williams
Jody Farr
Ed Kennedy
Vaughan Benson

## Table Of Contents

## List of Figures and Tables

# 1.0 INTRODUCTION

## 1.1 Background

An accurate and reliable way of measuring wave properties is fundamental for proper evaluation of ship hull designs and tank tests where specific types of waves are generated. The repeatability of wave probe data is essential for evaluating test results. These important qualities are in question for the current systems.

## 1.2 Problem

The wave probes use 30 gauge wire, which is quite thin. When in use for extended periods of time the wire can stretch. This causes drifting, linearity loss, and other problems. I will test lower gauge (thicker) wire with the original IOT wave probe conditioner.

IOT currently utilizes three wave probe signal conditioning circuits: the Branckner, original IOT, and modified IOT. The Branckner is a commercially made product, while the modified IOT is an updated version of the original IOT design, both of which were in house designs. The problem is that the IOT designs have problems with signal lagging and response time compared to the Branckner. I will address the problems by trying to develop a new system to capture the wave probe capacitance and convert it into usable data.

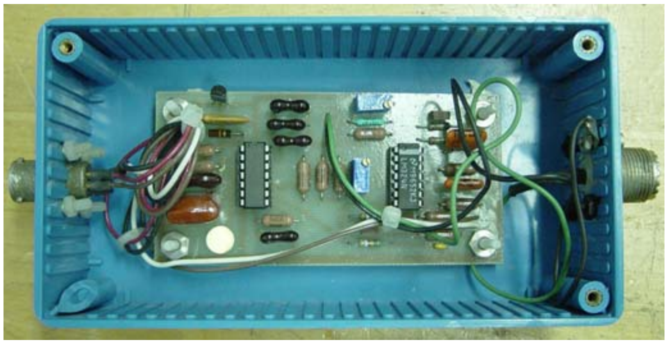


**Figure 1.1 Original IOT Wave Probe Conditioner**

## 1.3 Scope


**Figure 1.2 Wave Probes**

Previous studies were conducted on the wave probe signal conditioner boxes. Various problems were found with the systems under these tests. I will examine the findings from the studies and perform some tests of my own. After testing is complete, with the help of people in the electronics lab I will try to find a solution to the problems. If no solution can be found then I will look into an alternate method of conditioning the wave probe signals.

Lower gauges of wire will be ordered and tested under the original IOT design. The wires will be thicker and therefore should better withstand the stresses of testing conditions in the wave tank. These lower gauge wires also produce a higher range of capacitance per unit length, which leads to a greater signal output level change for the same wave height. If these wires produce linear and repeatable results, then the amount of time spent changing and re-stringing the wave probes can be reduced by using these wires as the standard for probe wiring.

The end goal is to have robust wave probes coupled with signal conditioning electronics which produces reliable and accurate data. The new wave probe signal conditioner design could become an important part of the wave probe data collection system at IOT's Tow Tank and OEB Facilities.

# 2.0 WIRE GAUGE TEST

## 2.1 Stringing the Probes

There does not appear to be a work instruction that describes the method used to string wave probes. It isn't very complicated, but to get it right takes practice. Correct wave probe stringing can be accomplished in a few easy steps:

1. Cut the correct length of wire that you need. The length is double the probe height, plus the amount of wire needed to make the connection.
2. Place an elastic band at the top end, between the grommet and metal post. Stretch out the elastic band with something about an Inch long. I used a pencil eraser.
3. Make one pass over the grommet on the bottom. Do not double wrap the wire, or make a knot on the bottom end. This causes capacitance and linearity issues, which are still to be studied.

**Figure 2.1 Bottom Grommet**

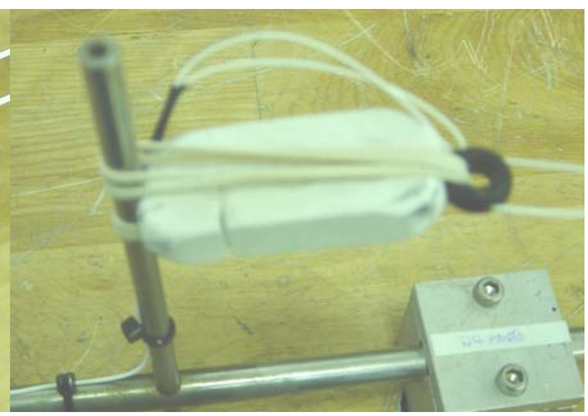


**Figure 2.2 Upper Grommet and Elastic Band**

4. Pass the wire through the top grommet, tying the two wire ends together in a knot. You can now remove the eraser, which allows the elastic band to take up the slack in the wire. The tension from the elastic band should be enough to make the wire taught, but not enough to damage it. It should still wobble a little when you touch it.

5. The ends of the wire can now be twisted together and soldered to the output wire. Make sure afterwards to tie the wire down so that it is not loose and floating around. It should be noted that any changes in wire position will change the capacitance.

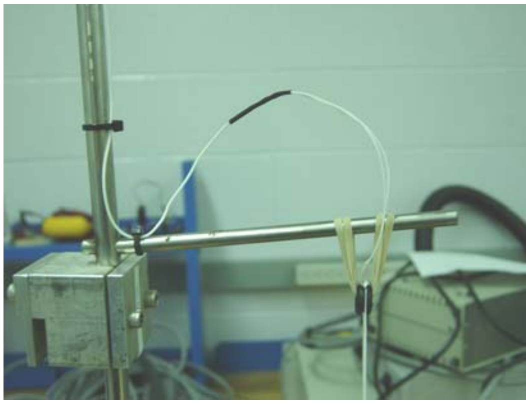


**Figure 2.3 Loop in wire (Wrong)**

**Figure 2.4 Wire Tied Down (Right)**

## 2.2 Test Outline

The purpose of the tests was to check and see if it is possible to use lower gauges (thicker) wire on the wave probes. Thicker wire would make the wave probes more versatile. The procedure I went about was to string the probes with 24, 26, and 30 AWG white colored kynar wrapping wire, following the above instructions. Previous studies have shown that only the white kynar wire made by OK Industries produces linear results. The wave probes have marks on them every 10 cm. I recorded the voltage out of the original IOT wave probe signal conditioner at each 10 cm mark as the probe was submerged in the portable water tank. See Figure 2.5 below. I also recorded the capacitance at the center of the probe, bottom, and highest point I could get. I did this for each wire gauge. After all the data was collected I plotted it in Excel, which can be seen in section 2.3.

## 2.3 Test Results

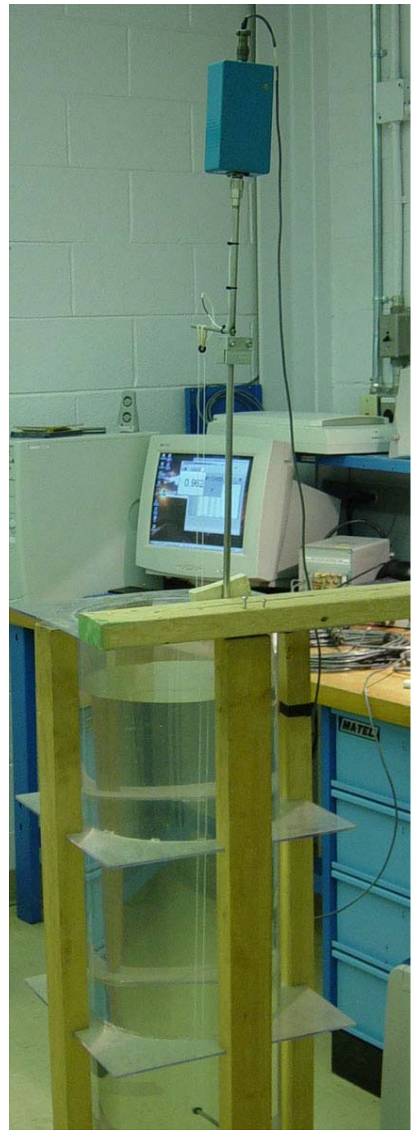


**Figure 2.5 Wave Tank**

The results below show that outputs with different gauge wires are linear and usable. The thicker wire produces a higher capacitance and consequently higher output voltage per unit length. The 30 gauge wire gives 7.57 mV/cm or 3.64 pF/cm, seen in Table 2.2 below. The 26 gauge gives 20 mV/cm or 10.86pF/cm, seen in Table 2.4. The 24 gauge wire gives 32.26 mV/cm or 17.86 pF/cm, seen in Table 2.6. These measurements were made over a 70 cm span. Any small errors most definitely came from human error, as I was only judging the probe marks by eye. This was a very simple test to see if it is even possible to use smaller gauge wires on the wave probes. Much more extensive tests would have to be done to verify its usability in tanks conditions. Future tests for example would need to take into account drag on the wire from waves. It is expected that the wave probe with a thicker 24 AWG wire would show a slightly different response than a wave probe with 30 AWG wire under the same wave conditions due to the increased drag. For the results below I measured capacitance at the highest point in the water, lowest point, and middle of the probe. NRC – CHC, has conducted many tests and they should be contacted.

### 2.3.1 30 Gauge Wire Test

| Distance (cm) | Voltage (V) | Capacitance (nF) |
|---|---|---|
| 40 | 0.14 | 0.655 |
| 50 | 0.07 | |
| 60 | 0 | 0.583 |
| 70 | -0.07 | |
| 80 | -0.14 | |
| 90 | -0.2 | |
| 100 | -0.27 | |
| 110 | -0.34 | 0.4 |
| Out of Water | -0.39 | 0.375 |

**Table 2.1 30 AWG Wire Data**

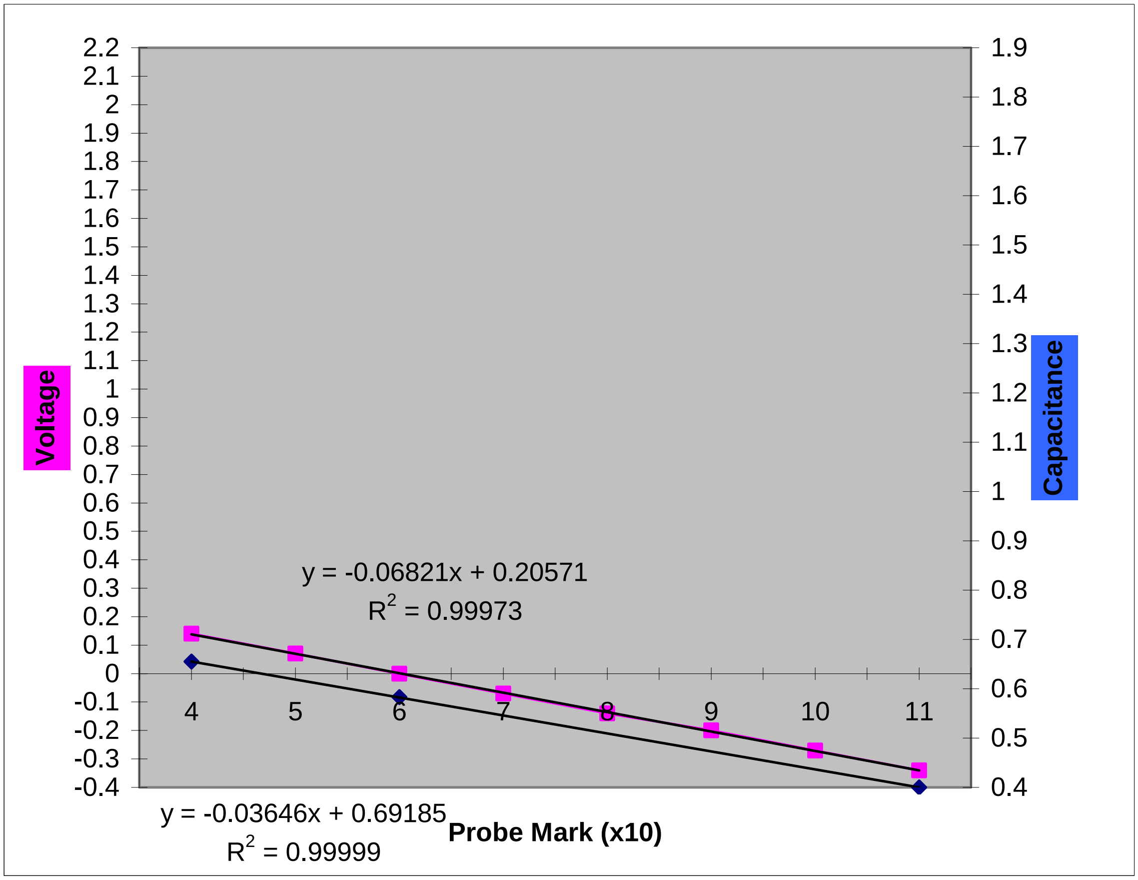



**Table 2.2 30 AWG Wire Graph**

### 2.3.2 26 Gauge Wire Test

| Probe Mark (cm) | Voltage (V) | Capacitance (nF) |
|---|---|---|
| 40 | 1.17 | 1.23 |
| 50 | 0.96 | |
| 60 | 0.768 | 1.01 |
| 70 | 0.568 | |
| 80 | 0.36 | |
| 90 | 0.17 | |
| 100 | -0.027 | |
| 110 | -0.23 | 0.47 |
| Out of Water | -0.375 | 0.38 |

**Table 2.3 26 AWG Wire Data**

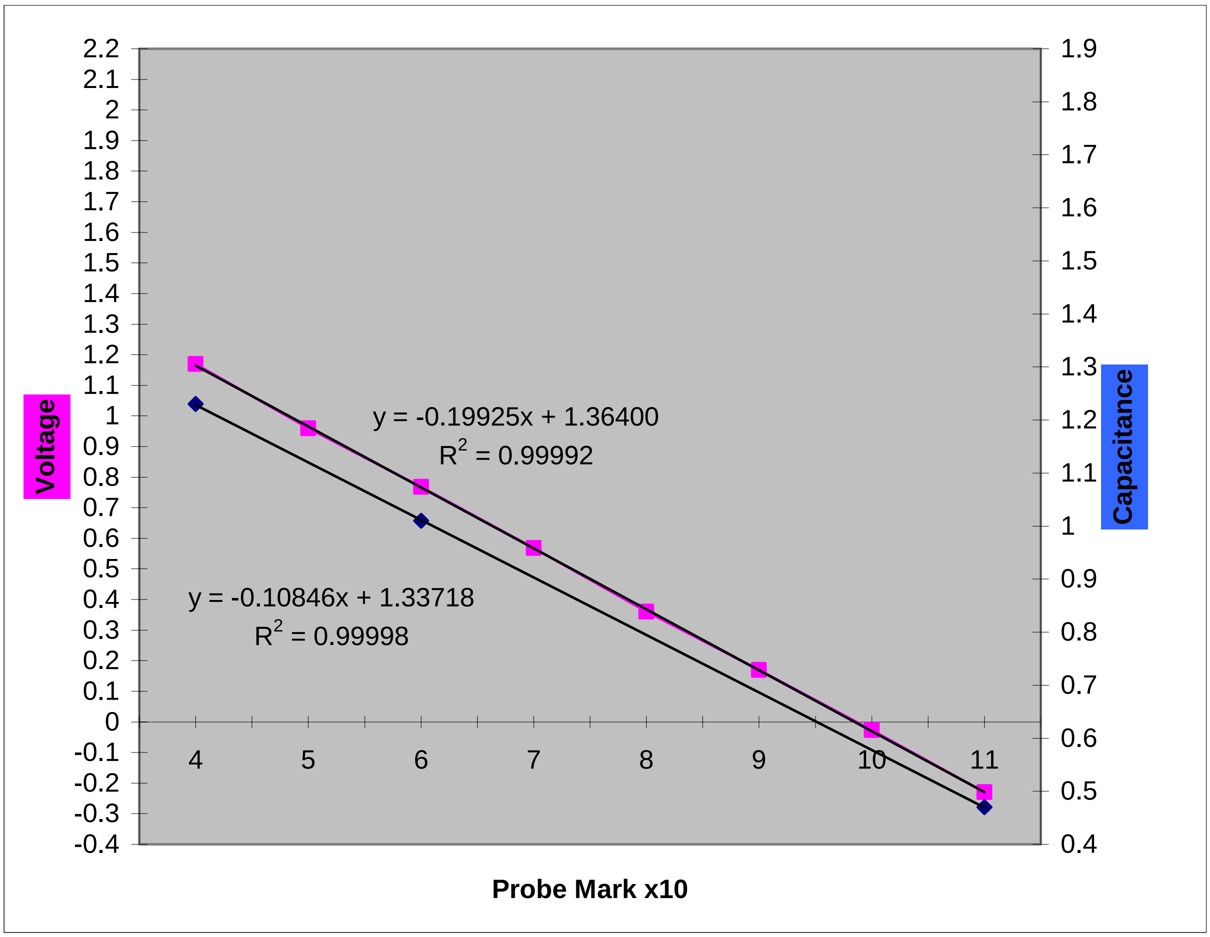



**Table 2.4 26 AWG Wire Graph**

### 2.3.3 24 Gauge Wire

| Probe Mark (cm) | Voltage (V) | Capacitance (V) |
|---|---|---|
| 40 | 2.055 | 1.75 |
| 50 | 1.73 | |
| 60 | 1.412 | 1.39 |
| 70 | 1.105 | |
| 80 | 0.793 | |
| 90 | 0.47 | |
| 100 | 0.15 | |
| 110 | -0.203 | 0.5 |
| Out of Water | -0.393 | 0.39 |

**Table 2.5 24 AWG Wire Data**

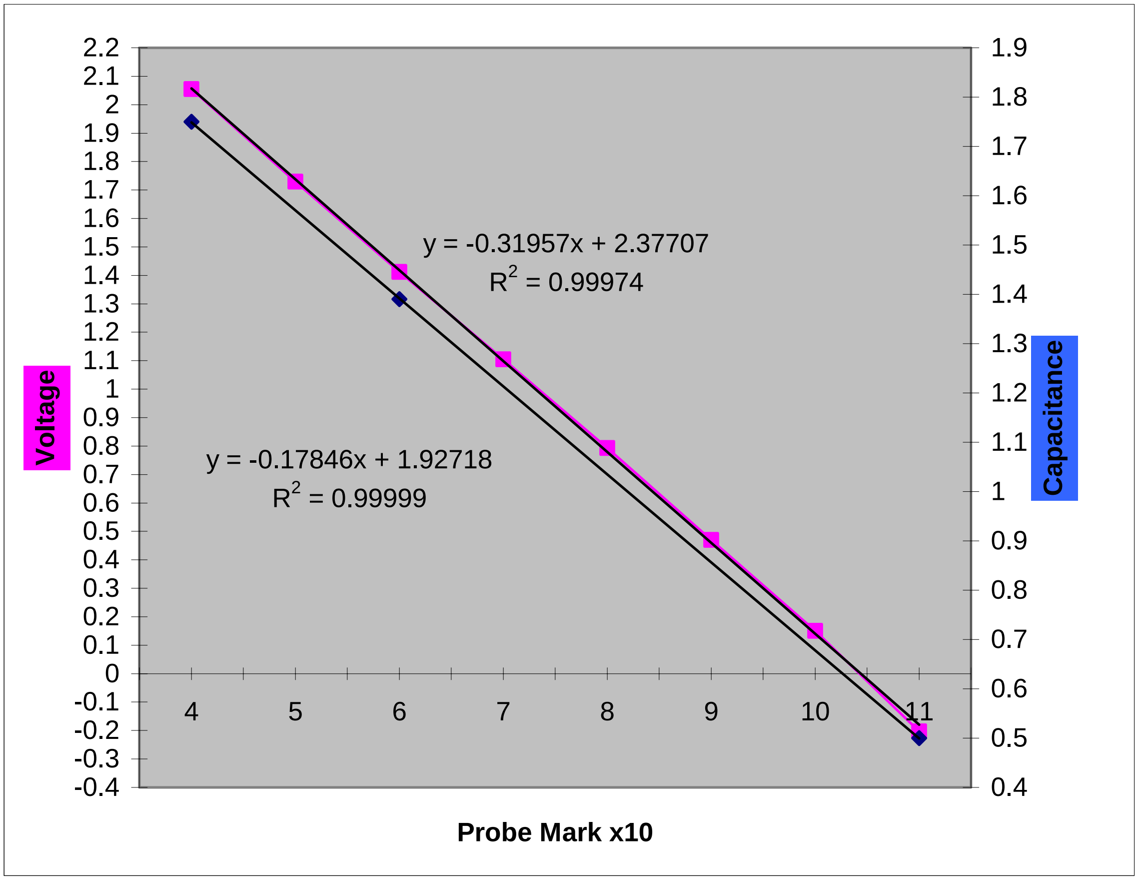



**Table 2.6 24 AWG Wire Graph**

# 3.0 WAVE PROBE CONDITIONER

## 3.1 Hardware

The IOT waveprobe conditioner design operates as follows. The capacitance of the wave probe is connected to the front end of a dual 555 timer (556). When the wave probe is submerged into water, its capacitance increases. This is due to the electrical properties of the wire and the probe. The rise in capacitance increases the frequency out of the first 555 due to the way it is configured. The second 555 timer is triggered by the output of the first 555 timer to make it operate in the "one shot" mode. The frequency that is output out of the timer then goes through a low-pass filter network, which converts the frequency into a DC voltage. Refer to Figure 3.2. Studies have shown that when compared to the Branckner model, the IOT design had problems with signal lagging, and doesn't reach its peak output voltage.

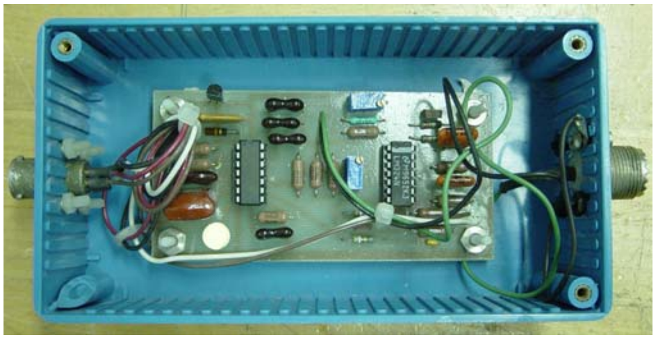


**Figure 3.1 Original IOT Wave Prove Conditioner**

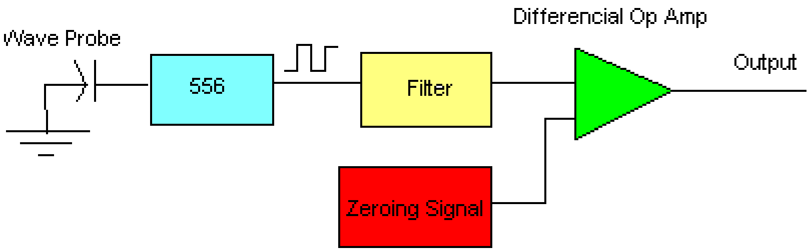



**Figure 3.2 Original IOT Wave Probe Conditioner Block Diagram**

What I set out to do first was find a better way to sample the 556 timer output and change it to a voltage. My plan was to use a Microchip PIC to take the output from the 556 timer and convert it to RS232 serial digital format, which was then sent to the Digital to analog converter (DAC). After some consideration we realized that it would not be advisable to redesign the current wave probe conditioners. This was because the current systems work, and there are new ways to accomplish the same task. So I decided to design a conditioner that could be used in a different data collection system. Instead of sending the data to a DAC, I sent it out over RS232 to a PC. The schematic for my circuit can be seen in Appendix A.

### 3.2 Software

There was already some PIC software developed by Greg Janes, which was used in the IOT developed RPM boards. It takes the rpm's from a motor in the form of a pulse train, samples it with a PIC, and outputs the data to a DAC which sends the signal to a data acquisition system. I modified the code, with Greg's help, to take in a frequency from a 555 timer, convert it to usable information, and transfer it out to a computer in RS232 format. Calculations were used to determine what value of resistors I required to get the frequency out of the 555 timer into the range that the PIC can use, these can be seen in Figure 3.2.

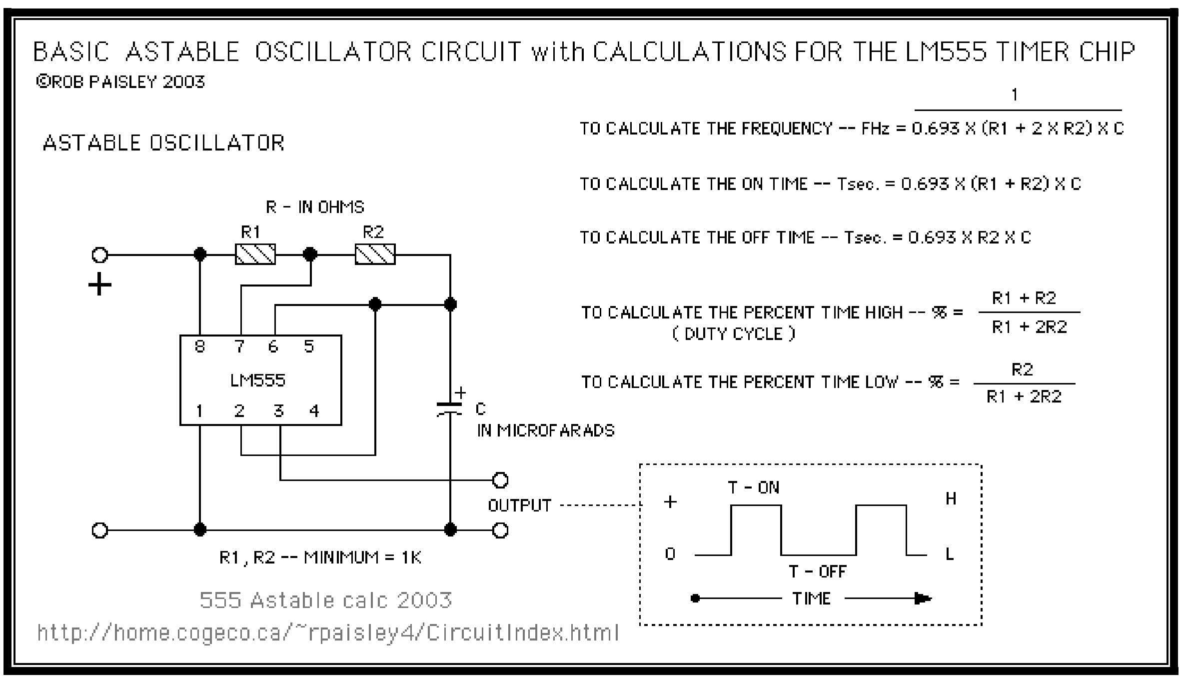


**Figure 3.3 LM555 configured for Astable Operation and Calculations**

I was new to the Hi-Tech compiler that Greg utilizes, so it took some time to learn the commands and format that the C++ code is written in. Then I proceeded to edit the code to suit my needs. I changed the code around significantly and managed to get it to work with RS-232. Then I tested the circuit by using HyperTerminal on the computer. The PC displayed unstable readings due to the serial port being unable to process data quickly enough. This was because the PIC has no flow control or handshaking. In order to get it working correctly I had to implement a static flow control in the PIC, which was just a delay to slow down transfers. The various changes can be seen in appendix B, where you can see the complete code.

### 3.3 Calibration Buttons

Another feature that makes this design unique is the addition of min and max calibration buttons. With the current wave probe conditioners, calibration is a long and tedious process. I have designed a way to make that process very simple. Using two buttons, digital calibration can be achieved.

When the wave probe is at its highest point in the water, you press the “Max” button. The PIC captures a reading at that point and saves it in memory. Then you put the wave probe at its lowest point in the water, and press the min button, which saves the reading. In the PIC code the wave probe input is scaled with the minimum and maximum values into a 16-bit number through some simple equations. The code can be seen below in Table 3.1.

```
dDeltaT = dTime - dTimeOld;
if (dDeltaT > dMax)
        dDeltaT = 4095;
else if (dDeltaT < dMin)
        dDeltaT = 0;
else{
        dDeltaT -= dMin;
        dDeltaT = (dDeltaT/dRange) * 4095;
}
cOut = (unsigned char) dDeltaT;
TxBuffer[1] = cOut;

else if (RBIF) {
        inputb = PORTB;
        if (inputb == 16)
                dMin = dSample;
        if (inputb == 32)
                dMax = dSample;
        dRange = dMax - dMin;
        RBIF = 0;
```

**Table 3.1 Min/Max Button Code**

## 3.4 Conditioner Testing

I tested my wave probe conditioner with two different wire gauges to get the two extremes of capacitance. The first test was done with 30 gauge wire and the second with 24 gauge. The results can be seen below. Note that the “output” is just a number that the PIC sends to HyperTerminal, it is derived from the timing inside of the PIC and doesn’t equate to an actual quantity. This is why I implemented the min and max feature, because then you can actually tell how deep the probe is. The linearity doesn’t change with capacitance either, which is a good thing.

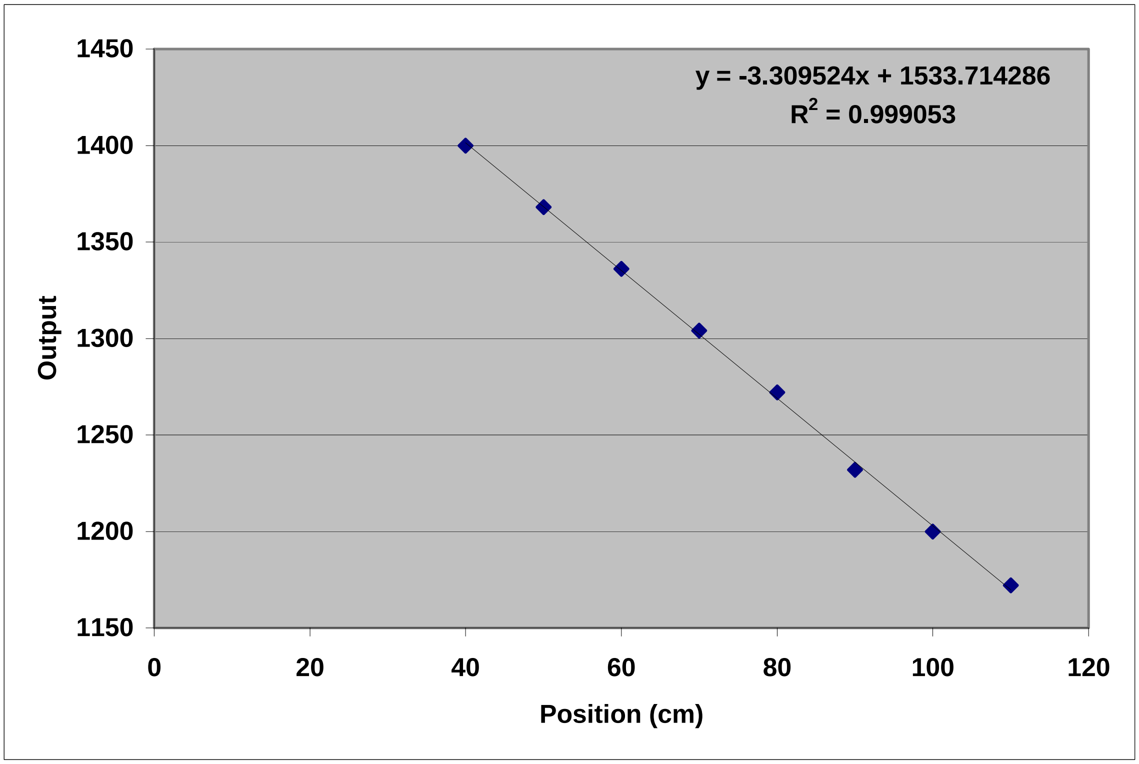



**Table 3.2 30 AWG Wire Test**

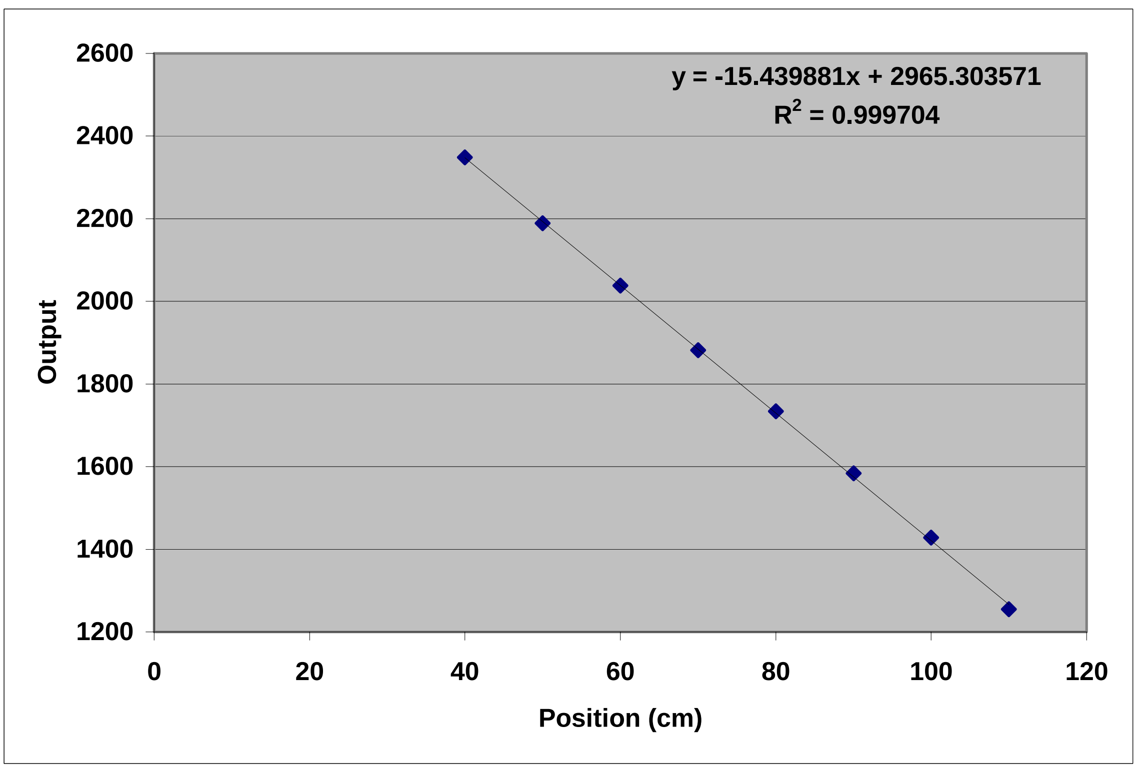



**Table 3.3 26 AWG Wire Test**

## 4.0 CONCLUSION

This project has been a success in a number of ways. Firstly for the amount of time spent working on it, this was definitely worth looking into. This wave probe conditioner has potential applications in many areas that involve taking a frequency and converting it data that is usable to a computer. IOT now has another potential source of conditioning for wave probes and signals of other types. The PIC is not only capable of receiving frequencies, but also analog signals. Digital systems are also a lot less susceptible to noise, and therefore work well in noisy environments. With digital signals you also skip the problematic analog signal conditioner portion of the system and feed the output directly into a computer, which is where all the data gets processed in the end. Development is underway to implement a server computer with the ability to take in data from multiple serial ports.

It should be noted that my tests of the different gauges of wire on the wave probes shows that the outputs are linear, and the system is possibly capable of using thicker wire. Any deficiencies seen are quite likely caused by human error. But in order to truly see how the wire reacts, it would need to be tested in actual test conditions at IOT's test Facilities or using a suitable controlled apparatus. The tests I did are merely basic tests to see if the wires would work at all.

## 5.0 RECOMMENDATIONS

This project is not completely finished due to time constraints. I have completed the designing, prototyping and part of the testing of the idea. There are a few more things that would have to be completed before it is a finished product. One such thing would be to package the circuit into an enclosure, and then decide where it would derive the 5 volts required to operate.

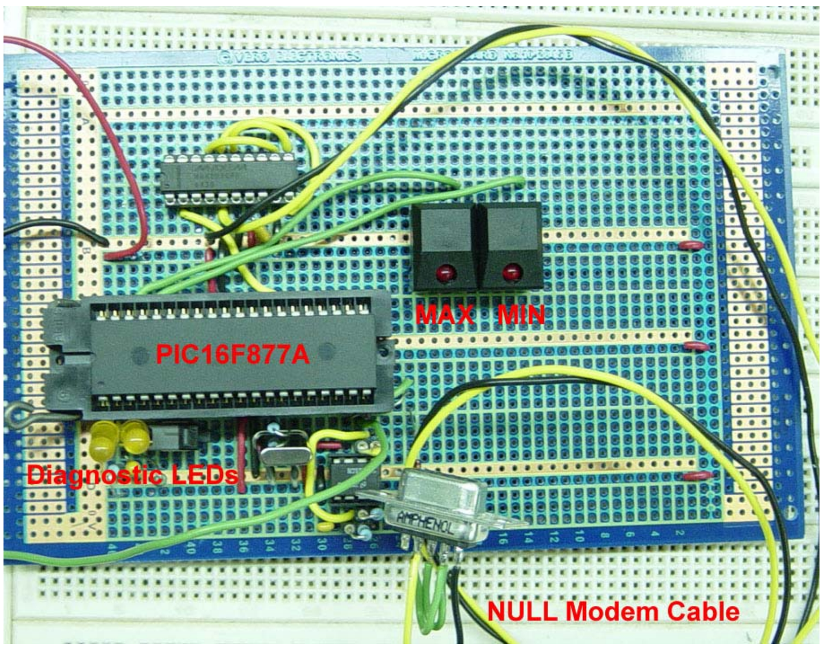



**Figure 5.1 Prototyped Wave Probe Conditioner**

It also needs to be tested with Keith Mew's server software. The PIC outputs the correct format, but I haven't tested to see if it works. Keith's server would need to be tested with the conditioner to see if the transfer rate is too high, and the data format is correct. Because the server multiplexes multiple inputs into one serial line, if the transfer rate of the PIC is too high, you start to use up the bandwidth of the serial output. Changing a couple of delays in the PIC code can solve this problem. This is good because when you slow down the transfer rate of the PIC you also increase consistency of the PIC output data.

More precise tests need to be done on the conditioner to see how it reacts under typical conditions, and to see how well it performs. This would involve either testing the conditioner in actual tank settings, or testing it with a 3-axis linear table arrangement, as discussed in Justin Wheeler's report, The Dynamic Response of Capacitance Wave Probes.

Jacque Daze of NRC's Canadian Hydraulic Center should be contacted about the wire gauge tests. CHC has completed an extensive number of tests with different wave probe wire gauges, as well as dynamic response tests, and have results that they are willing to share. I have contacted Jacques, and he sent me a lot of data, which was gathered using the commercially available Brancker WG-30 wave probe signal conditioner.

## 6.0 References

[Anonymous] Microchip: PIC16F87XA Datasheet (2003). Available from: http://ww1.microchip.com/downloads/en/DeviceDoc/39582b.pdf via the Internet. Accessed February 18, 2005.

[Anonymous] Maxim Integrated Products: ICM7555/7556 (1994). Available from http://pdfserv.maxim-ic.com/en/ds/ICM7555-ICM7556.pdf via the Internet. Accessed February 17, 2005.

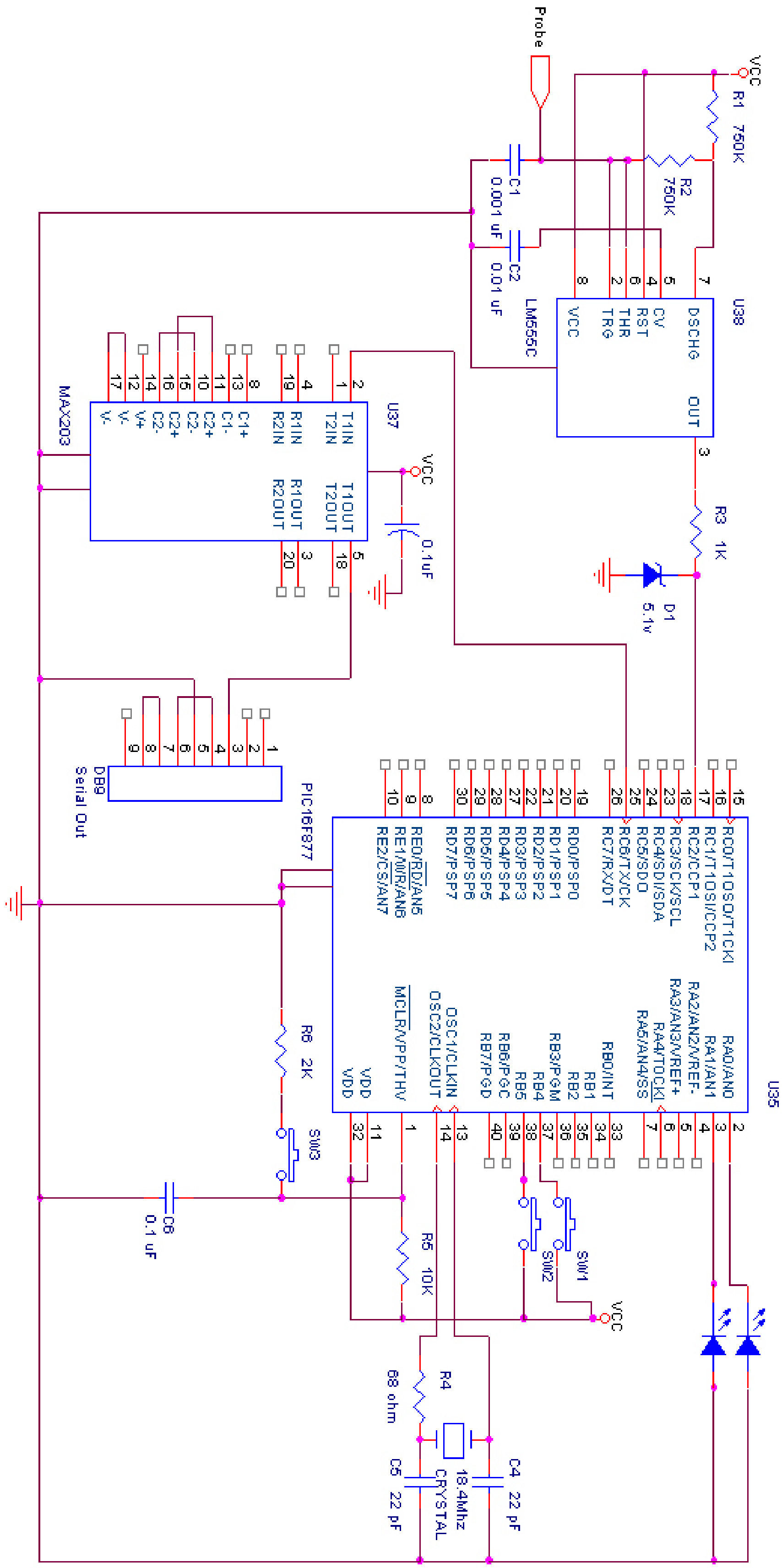
[Anonymous] Maxim Integrated Products: MAX203 (2003). Available from http://pdfserv.maxim-ic.com/en/ds/MAX200-MAX213.pdf via the Internet. Accessed March 3, 2005.

Paisley, R. LM555 Circuits (2004). Available from: http://home.cogeco.ca/~rpaisley4/LM555.html#3 via the Internet. Accessed February 18, 2005.

Sheridan, K. A digital water level meter (2002); Available from: http://hawthorn.csse.monash.edu.au/~njh/electronics/watersensor/ via the Internet. Accessed February 25, 2005.

Wheeler, J (2004). The Dynamic Response of Capacitance Wave Probes. St. John's: National Research Council's Institute for Ocean Technology.

# APPENDIX A: Wave Probe Conditioner Circuit


Probe
VCC
R1 750K
R2 750K
C1 0.001 uF
C2 0.01 uF
U38
LM555C
R3 1K
D1 5.1v
U37
MAX203
0.1uF
DB9
Serial Out
U35
PIC16F877
R6 2K
SW3
C6 0.1 uF
R5 10K
SW1
SW2
R4 68 ohm
C5 22 pF
C4 22 pF
18.4Mhz
CRYSTAL

# APPENDIX B: PIC Source Code

```
/*-----------------------------------------------------------------------------
 * Wave Probe Signal Conditioner
 *
 * Ver 1.2: Chad Collett: April 21, 2005
 * This is the Basic code, it doesn't contain limit switches, and is in a format
 * that HyperTerminal can read. It is not in Keith Mew's server configuration.
 * This code outputs the raw timing of DdeltaT to the serial port, this is a number
 * determined by the constantly counting time inside the PIC.
 *-----------------------------------------------------------------------------*/

#include <pic.h>
#include <stdio.h>

/* Function Prototypes */
void InitParallelPorts();
void InitCCPTimer1();
void InitUART();
void interrupt ISR();

/* Globals */
unsigned int iTMR1Roll;         // Count TMR1 roll overs (in ISR) to extend it to 32 bits.
unsigned int iTMR1;
unsigned char iCCP1H;           // Capture high.
unsigned char iCCP1L;           // Capture low.
unsigned char iGotcha;
unsigned int bFirst;
unsigned int iSkip;

//double dSample;
unsigned int iCount;

bank1 double dTime;
bank1 double dTimeOld;
bank1 double dDeltaT;

int i, j;

// ---------------------------------
main () {
// ---------------------------------

        InitParallelPorts();  //25 Sept 2003: RD3 is an output

        RA0 = 1;                // Indicate running, then indicates edge detection.

        InitUART();
        InitCCPTimer1();

        bFirst = 1;
        iGotcha = 0;      // Flag to show when new capture occurs.
        iTMR1Roll = 0;
        iTMR1 = 0;
        iCCP1H = 0;
        iCCP1L = 0;
        iSkip = 0;
```

```
    dTime = 0.0;
    dTimeOld = 0.0;
    dDeltaT = 0.0;

    // Setup the Interrupts:
    RCIE = 0;

    CREN = 0;       // Clear and then set the CREN bit to clear
    CREN = 1;       // OERR and FERR (if any) and enable async reception.

    // Enable Timer1 interrupts
    TMR1IE = 1;

    // Enable interupts
    GIE = 1;// Global Interupt Enable bit

    iGotcha = 0;
    CCP1IF = 0;
    CCP1IE = 1; // Enable capture.

    i = 0;
    j = 0;

    while (1) {

        if (iGotcha == 1) {

            dTime = ((double)(iTMR1) * 65536.0);
            dTime += ((double)(iCCP1H) * 256.0);
            dTime += (double)(iCCP1L);

            iGotcha = 0;        // Release the buffer variables to the ISR.

            if (bFirst == 1) {
                bFirst = 0;
            }
            else {
                dDeltaT = dTime - dTimeOld;         // dDeltaT is in counts.

                if (iSkip > 749) {  // Only output every 749 signal

                    iCount = (unsigned int) dDeltaT;
                    printf("%d\r",iCount);
                    iSkip = 0;
                }
            } // end else
            dTimeOld = dTime; // Save time for next iteration.
        } // end if iGotcha == 1
    } // while (1)
} // main

// -------------------------------
void InitParallelPorts() {
// -------------------------------
    // PORTA
    ADCON1 = 0x00;          // Set 0-7 pins of port A as analog to digital converters.
```

```
          PORTA = 0x00;           // Default output pins to low.
          TRISA = 0x00;           // Set 0-7 pins of port A to output.
        // PORTB
        PORTB = 0x00;           // Default output pins to low.
        TRISB = 0xFF;           // Set 0-7 pins of port B as input.
        PORTC = 0x00;           // Default outputs to low.
        TRISC = 0x96;           // 1001 0110
        // PORTD
        PORTD = 0x00;           // Default values for outputs.
        TRISD = 0x06;           // Set RD2 and RD1 to input.
        // PORTE
        PORTE = 0x00;           // Default outputs to low.
        TRISE = 0x00;           // Set 0-7 pins of port E as output.
}

// -------------------------
void InitCCPTimer1() {
// -------------------------
        CCP1CON = 0x05;                 // capture every rising edge.
        CCP2CON = 0x05;

        TMR1IF = 0;                     // Clear the interrupt flag directly (i.e. no clr on read)
        T1CON = 0x31;           // 1:8 prescale, internal Fosc/4, start.
        iTMR1Roll = 0;          // Count the roll overs in the ISR to extend the clock to 32 bits.
        //TMR1IE = 1;                   // Enable timer interrupt in main routine before starting
measure.
}

// ------------------------
void InitUART() {
// ------------------------
        SPBRG = 9;              // 115.2k

        RCIF = 0;               // reset flag
        BRGH = 1;               // high baud rate
        SYNC = 0;               // asynchronous
        SPEN = 1;               // enable serial port pins
        CREN = 1;               // enable reception
        SREN = 0;               // no effect
        TXIE = 0;               // disable tx interrupts
        TX9  = 0;               // 8-bit transmission
        RX9  = 0;               // 8-bit reception
        TXEN = 1;               // enable the transmitter
        PEIE = 1;               // enable peripheral interrupts
        PIE1 = 0;               // clear all peripheral interrupt enables
        RCIE = 0;               // (Enable it when ready to use.)
}

// Used by printf to transmit byte
// --------------------------------------
void putch(unsigned char byte){
// --------------------------------------
        unsigned char stat;
        stat = TXSTA;
        if ((stat & 0x02) != 0){
                TXREG = byte;
```

```
				for (i=0;i<99;i++) {}
				i = 0;
		}
}

// ------------------------------------ */
void interrupt ISR() {
// ------------------------------------
		if (TMR1IF) {
				++iTMR1Roll;									// Increment the
roll counter.
				if (iTMR1Roll > 2637) {
						iTMR1Roll = 0;							// Don't let the
counter cause overflow error.
						bFirst = 1;
				}
				iSkip++; //Increment iSkip
				TMR1IF = 0;			// Clear the interrupt flag directly (i.e. no clr on read)
		}

		else if (CCP1IF) {
						if (iGotcha == 1) {
								RA0 = 1;			// Indicate overrun.
						}
						else {
								iTMR1 = iTMR1Roll;
								iCCP1H = CCPR1H;
								iCCP1L = CCPR1L;
								iGotcha = 1;	// Let the main know we have a new capture.
								RA0 = 0;			// Most recent capture did not cause
overrange.
						}
						CCP1IF = 0;			// Clear the interrupt flag.
		} // if CCP1IF
}
```

```
/*----------------------------------------------------------------------------
 * Wave Probe Signal Conditioner
 *
 * Development History:
 *
 * Ver 1.3: Chad Collett: Mar 11, 2005
 * This code is optimised for Keith Mew's Server. It contains the code for
 * the limit buttons that I created, as well as the scaling to put the
 * data into a 12 bit word.
 *---------------------------------------------------------------------------*/

#include <pic.h>
#include <stdio.h>

/* Function Prototypes */

void InitParallelPorts();
void InitCCPTimer1();
void InitUART();
void interrupt ISR();

/* Globals */

unsigned int iTMR1Roll;          // Count TMR1 roll overs (in ISR) to extend it to 32 bits.
unsigned int iTMR1;
unsigned char iCCP1H;            // Capture high.
unsigned char iCCP1L;            // Capture low.
unsigned char iGotcha;
unsigned int bFirst;
unsigned int iInactTMR;          // Inactivity time out timer. Incremented in TMR1 ISR.
unsigned int iSkip;

bank1 double dTime;
bank1 double dTimeOld;
bank1 double dDeltaT;

double dSample;
unsigned char cOut;

int inputb;
double dMin;
double dMax;
double dRange;

#define BOT 0xAAAA
#define EOT 0xBBBB

unsigned int TxBuffer[4];
unsigned char *tptr;
unsigned char txidx;

// ---------------------------------
main () {
// ---------------------------------

        InitParallelPorts();  //25 Sept 2003: RD3 is an output
```

```
  RA0 = 1;                    // Indicate running, then indicates edge detection.

InitUART();
InitCCPTimer1();

tptr = (unsigned char *)TxBuffer; //Point the byte pointer to transmit buffer.
txidx = 0;
TxBuffer[0] = BOT;
TxBuffer[3] = EOT;

bFirst = 1;
iGotcha = 0;       // Flag to show when new capture occurs.
iTMR1Roll = 0;
iInactTMR = 0;
iTMR1 = 0;
iCCP1H = 0;
iCCP1L = 0;
iSkip = 0;

dTime = 0.0;
dTimeOld = 0.0;
dDeltaT = 0.0;

// Setup the Interrupts:
RCIE = 0;

CREN = 0;          // Clear and then set the CREN bit to clear
CREN = 1;          // OERR and FERR (if any) and enable async reception.

// Enable Timer1 interrupts
TMR1IE = 1;

// Enable interupts
GIE = 1;// Global Interupt Enable bit

// RB Interrupts, for the calibration buttons
RBIE = 1;
RBIF = 0;

iGotcha = 0;
CCP1IF = 0;
CCP1IE = 1; // Enable capture.

dMin = 1000;
dMax = 2500;
dRange = 1500;

// Loop Forever:
while (1) {

        if (iGotcha == 1) {
                dTime = ((double)(iTMR1) * 65536.0);
                dTime += ((double)(iCCP1H) * 256.0);
                dTime += (double)(iCCP1L);
```

```
                iGotcha = 0;        // Release the buffer variables to the ISR.

                if (bFirst == 1) {
                        bFirst = 0;
                }
                else {
                        dDeltaT = dTime - dTimeOld;                 // dDeltaT is in counts.

                        if (iSkip > 599) {  // Only output every 500th sample, so that the
transmitter isn't overrun
                                dSample = dDeltaT;
                                if (dDeltaT > dMax)
                                        dDeltaT = 4095;
                                else if (dDeltaT < dMin)
                                        dDeltaT = 0;
                                else{
                                        dDeltaT -= dMin;
                                        dDeltaT = (dDeltaT/dRange) * 4095;
                                }
                                cOut = (unsigned char) dDeltaT;
                                TxBuffer[1] = cOut;
                                TxBuffer[2] = cOut;
                                TXIE = 1;
                                TXEN = 1;
                                iSkip = 0;
                        }
                } // end else
                dTimeOld = dTime; // Save time for next iteration.
        } // end if iGotcha == 1

        iSkip++; //Increment iSkip

    } // while (1)
} // main

// -------------------------------
void InitParallelPorts() {
// ------------------------------
    // PORTA
    ADCON1 = 0x00;                  // Set 0-7 pins of port A as analog to digital converters.
    PORTA = 0x00;           // Default output pins to low.
    TRISA = 0x00;           // Set 0-7 pins of port A to output.

    // PORTB
    PORTB = 0x00;           // Default output pins to low.
    TRISB = 0xFF;           // Set 0-7 pins of port B as input.

    PORTC = 0x00;           // Default outputs to low.
    TRISC = 0x96;           // 1001 0110

    // PORTD
    PORTD = 0x00;           // Default values for outputs.
    TRISD = 0x06;           // Set RD2 and RD1 to input.

    // PORTE
    PORTE = 0x00;           // Default outputs to low.
```

```
        TRISE = 0x00;              // Set 0-7 pins of port E as output.
  }

// -------------------------
void InitCCPTimer1() {
// -------------------------
        CCP1CON = 0x05;                  // capture every rising edge.
        CCP2CON = 0x05;

        TMR1IF = 0;                      // Clear the interrupt flag directly (i.e. no clr on read)
        T1CON = 0x31;              // 1:8 prescale, internal Fosc/4, start.
        iTMR1Roll = 0;             // Count the roll overs in the ISR to extend the clock to 32 bits.
        //TMR1IE = 1;                    // Enable timer interrupt in main routine before starting
measure.
}

// ------------------------
void InitUART() {
// ------------------------
        SPBRG = 9;                 // 115.2k

        RCIF = 0;                  // reset flag
        BRGH = 1;                  // high baud rate
        SYNC = 0;                  // asynchronous
        SPEN = 1;                  // enable serial port pins
        CREN = 1;                  // enable reception
        SREN = 0;                  // no effect
        TXIE = 0;                  // disable tx interrupts
        TX9  = 0;                  // 8-bit transmission
        RX9  = 0;                  // 8-bit reception
        TXEN = 1;                  // enable the transmitter
        PEIE = 1;                  // enable peripheral interrupts
        PIE1 = 0;                  // clear all peripheral interrupt enables
        RCIE = 0;                  // (Enable it when ready to use.)
}

void interrupt ISR() {
// ------------------------------------

        if (TMR1IF) {
                ++iTMR1Roll;                                         // Increment the
roll counter.
                //if (iTMR1Roll > 50000) {
                if (iTMR1Roll > 2637) {
                //if (iTMR1Roll > 528) {
                        iTMR1Roll = 0;                               // Don't let the
counter cause overflow error.

        // Reset periodically to prevent cumulative

        // math problems.
                        bFirst = 1;
                }

                TMR1IF = 0;              // Clear the interrupt flag directly (i.e. no clr on read)
        }
```

```
	else if (CCP1IF) {
			if (iGotcha == 1) {
				// Overrun. The main routine hasn't finished
				// processing the previous capture.
				RA1 = 1;			// Indicate overrun.
			}
			else {
				iTMR1 = iTMR1Roll;
				iCCP1H = CCPR1H;
				iCCP1L = CCPR1L;
				iGotcha = 1;	// Let the main know we have a new capture.
				RA1 = 0;			// Most recent capture did not cause
overrange.
			}

			iInactTMR = 0;	// Clear the inactivity timer.
			CCP1IF = 0;			// Clear the interrupt flag.
	} // if CCP1IF

	else if (RBIF) {
		inputb = PORTB;
		if (inputb == 16)
			dMin = dSample;
		if (inputb == 32)
			dMax = dSample;
		dRange = dMax - dMin;
		RBIF = 0;
	}
	else if (TXIF) {
		if (txidx >= 8) { //8 Bytes to transmit
			txidx = 0;
			TXIE = 0;
		}
		else {
			int i;
			TXREG =(tptr[txidx]);
			for (i=0;i<50;i++) {}
			txidx++;
		}
	}
}
```

# APPENDIX C: Component Data Sheets

# PIC16F87X Data Sheet

## 28/40-Pin 8-Bit CMOS FLASH Microcontrollers

# PIC16F87X

## 28/40-Pin 8-Bit CMOS FLASH Microcontrollers

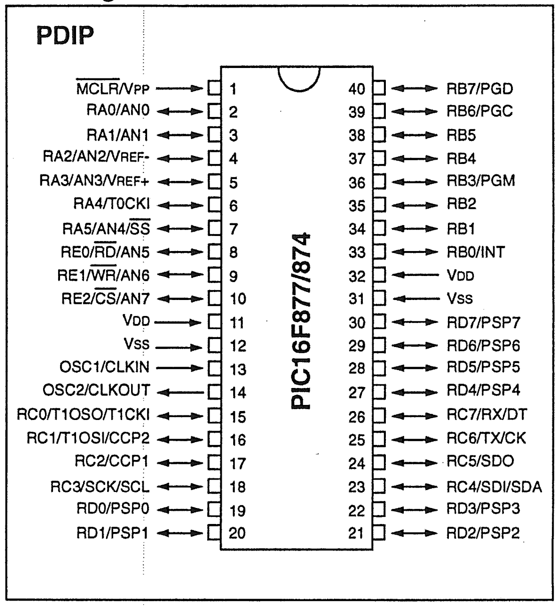
### Devices Included in this Data Sheet:

- PIC16F873
- PIC16F874
- PIC16F876
- PIC16F877

### Microcontroller Core Features:

- High performance RISC CPU
- Only 35 single word instructions to learn
- All single cycle instructions except for program branches which are two cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory,
  Up to 368 x 8 bytes of Data Memory (RAM)
  Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to the PIC16C73B/74B/76/77
- Interrupt capability (up to 14 sources)
- Eight level deep hardware stack
- Direct, indirect and relative addressing modes
- Power-on Reset (POR)
- Power-up Timer (PWRT) and
  Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving SLEEP mode
- Selectable oscillator options
- Low power, high speed CMOS FLASH/EEPROM technology
- Fully static design
- In-Circuit Serial Programming™ (ICSP) via two pins
- Single 5V In-Circuit Serial Programming capability
- In-Circuit Debugging via two pins
- Processor read/write access to program memory
- Wide operating voltage range: 2.0V to 5.5V
- High Sink/Source Current: 25 mA
- Commercial, Industrial and Extended temperature ranges
- Low-power consumption:
  - < 0.6 mA typical @ 3V, 4 MHz
  - 20 µA typical @ 3V, 32 kHz
  - < 1 µA typical standby current

### Pin Diagram




### Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during SLEEP via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- 10-bit multi-channel Analog-to-Digital converter
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I²C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) 8-bits wide, with external RD, WR and CS controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

## Pin Diagrams

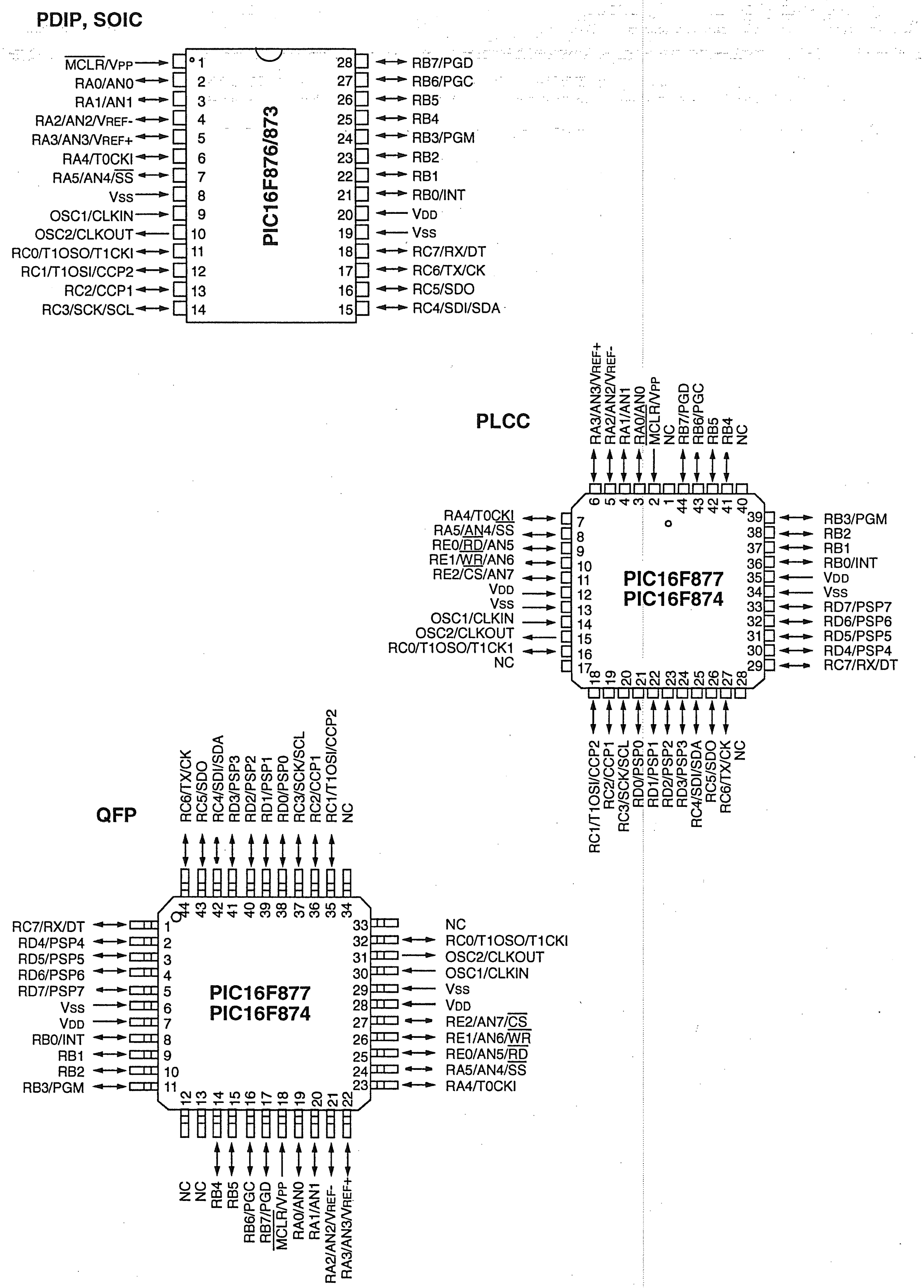


**TABLE 1-1: PIC16F873 AND PIC16F876 PINOUT DESCRIPTION**

| Pin Name | DIP Pin# | SOIC Pin# | I/O/P Type | Buffer Type | Description |
|---|---|---|---|---|---|
| OSC1/CLKIN | 9 | 9 | I | ST/CMOS(3) | Oscillator crystal input/external clock source input. |
| OSC2/CLKOUT | 10 | 10 | O | — | Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, the OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate. |
| $\overline{MCLR}$/VPP | 1 | 1 | I/P | ST | Master Clear (Reset) input or programming voltage input. This pin is an active low RESET to the device. |
| | | | | | PORTA is a bi-directional I/O port. |
| RA0/AN0 | 2 | 2 | I/O | TTL | RA0 can also be analog input0. |
| RA1/AN1 | 3 | 3 | I/O | TTL | RA1 can also be analog input1. |
| RA2/AN2/VREF- | 4 | 4 | I/O | TTL | RA2 can also be analog input2 or negative analog reference voltage. |
| RA3/AN3/VREF+ | 5 | 5 | I/O | TTL | RA3 can also be analog input3 or positive analog reference voltage. |
| RA4/T0CKI | 6 | 6 | I/O | ST | RA4 can also be the clock input to the Timer0 module. Output is open drain type. |
| RA5/$\overline{SS}$/AN4 | 7 | 7 | I/O | TTL | RA5 can also be analog input4 or the slave select for the synchronous serial port. |
| | | | | | PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs. |
| RB0/INT | 21 | 21 | I/O | TTL/ST(1) | RB0 can also be the external interrupt pin. |
| RB1 | 22 | 22 | I/O | TTL | |
| RB2 | 23 | 23 | I/O | TTL | |
| RB3/PGM | 24 | 24 | I/O | TTL | RB3 can also be the low voltage programming input. |
| RB4 | 25 | 25 | I/O | TTL | Interrupt-on-change pin. |
| RB5 | 26 | 26 | I/O | TTL | Interrupt-on-change pin. |
| RB6/PGC | 27 | 27 | I/O | TTL/ST(2) | Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming clock. |
| RB7/PGD | 28 | 28 | I/O | TTL/ST(2) | Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming data. |
| | | | | | PORTC is a bi-directional I/O port. |
| RC0/T1OSO/T1CKI | 11 | 11 | I/O | ST | RC0 can also be the Timer1 oscillator output or Timer1 clock input. |
| RC1/T1OSI/CCP2 | 12 | 12 | I/O | ST | RC1 can also be the Timer1 oscillator input or Capture2 input/Compare2 output/PWM2 output. |
| RC2/CCP1 | 13 | 13 | I/O | ST | RC2 can also be the Capture1 input/Compare1 output/ PWM1 output. |
| RC3/SCK/SCL | 14 | 14 | I/O | ST | RC3 can also be the synchronous serial clock input/output for both SPI and $I^2C$ modes. |
| RC4/SDI/SDA | 15 | 15 | I/O | ST | RC4 can also be the SPI Data In (SPI mode) or data I/O ($I^2C$ mode). |
| RC5/SDO | 16 | 16 | I/O | ST | RC5 can also be the SPI Data Out (SPI mode). |
| RC6/TX/CK | 17 | 17 | I/O | ST | RC6 can also be the USART Asynchronous Transmit or Synchronous Clock. |
| RC7/RX/DT | 18 | 18 | I/O | ST | RC7 can also be the USART Asynchronous Receive or Synchronous Data. |
| VSS | 8, 19 | 8, 19 | P | — | Ground reference for logic and I/O pins. |
| VDD | 20 | 20 | P | — | Positive supply for logic and I/O pins. |

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

**Note 1:** This buffer is a Schmitt Trigger input when configured as the external interrupt.
**2:** This buffer is a Schmitt Trigger input when used in Serial Programming mode.
**3:** This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

**TABLE 1-2: PIC16F874 AND PIC16F877 PINOUT DESCRIPTION**

| Pin Name | DIP Pin# | PLCC Pin# | QFP Pin# | I/O/P Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| OSC1/CLKIN | 13 | 14 | 30 | I | ST/CMOS(4) | Oscillator crystal input/external clock source input. |
| OSC2/CLKOUT | 14 | 15 | 31 | O | — | Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate. |
| $\overline{\text{MCLR}}$/VPP | 1 | 2 | 18 | I/P | ST | Master Clear (Reset) input or programming voltage input. This pin is an active low RESET to the device. |
| | | | | | | PORTA is a bi-directional I/O port. |
| RA0/AN0 | 2 | 3 | 19 | I/O | TTL | RA0 can also be analog input0. |
| RA1/AN1 | 3 | 4 | 20 | I/O | TTL | RA1 can also be analog input1. |
| RA2/AN2/VREF- | 4 | 5 | 21 | I/O | TTL | RA2 can also be analog input2 or negative analog reference voltage. |
| RA3/AN3/VREF+ | 5 | 6 | 22 | I/O | TTL | RA3 can also be analog input3 or positive analog reference voltage. |
| RA4/T0CKI | 6 | 7 | 23 | I/O | ST | RA4 can also be the clock input to the Timer0 timer/counter. Output is open drain type. |
| RA5/$\overline{\text{SS}}$/AN4 | 7 | 8 | 24 | I/O | TTL | RA5 can also be analog input4 or the slave select for the synchronous serial port. |
| | | | | | | PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs. |
| RB0/INT | 33 | 36 | 8 | I/O | TTL/ST(1) | RB0 can also be the external interrupt pin. |
| RB1 | 34 | 37 | 9 | I/O | TTL | |
| RB2 | 35 | 38 | 10 | I/O | TTL | |
| RB3/PGM | 36 | 39 | 11 | I/O | TTL | RB3 can also be the low voltage programming input. |
| RB4 | 37 | 41 | 14 | I/O | TTL | Interrupt-on-change pin. |
| RB5 | 38 | 42 | 15 | I/O | TTL | Interrupt-on-change pin. |
| RB6/PGC | 39 | 43 | 16 | I/O | TTL/ST(2) | Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming clock. |
| RB7/PGD | 40 | 44 | 17 | I/O | TTL/ST(2) | Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming data. |

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

**Note 1:** This buffer is a Schmitt Trigger input when configured as an external interrupt.
**2:** This buffer is a Schmitt Trigger input when used in Serial Programming mode.
**3:** This buffer is a Schmitt Trigger input when configured as general purpose I/O and a TTL input when used in the Parallel Slave Port mode (for interfacing to a microprocessor bus).
**4:** This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

## TABLE 1-2: PIC16F874 AND PIC16F877 PINOUT DESCRIPTION (CONTINUED)

| Pin Name | DIP Pin# | PLCC Pin# | QFP Pin# | I/O/P Type | Buffer Type | Description |
|---|---|---|---|---|---|---|
| | | | | | | PORTC is a bi-directional I/O port. |
| RC0/T1OSO/T1CKI | 15 | 16 | 32 | I/O | ST | RC0 can also be the Timer1 oscillator output or a Timer1 clock input. |
| RC1/T1OSI/CCP2 | 16 | 18 | 35 | I/O | ST | RC1 can also be the Timer1 oscillator input or Capture2 input/Compare2 output/PWM2 output. |
| RC2/CCP1 | 17 | 19 | 36 | I/O | ST | RC2 can also be the Capture1 input/Compare1 output/PWM1 output. |
| RC3/SCK/SCL | 18 | 20 | 37 | I/O | ST | RC3 can also be the synchronous serial clock input/ output for both SPI and $I^2C$ modes. |
| RC4/SDI/SDA | 23 | 25 | 42 | I/O | ST | RC4 can also be the SPI Data In (SPI mode) or data I/O ($I^2C$ mode). |
| RC5/SDO | 24 | 26 | 43 | I/O | ST | RC5 can also be the SPI Data Out (SPI mode). |
| RC6/TX/CK | 25 | 27 | 44 | I/O | ST | RC6 can also be the USART Asynchronous Transmit or Synchronous Clock. |
| RC7/RX/DT | 26 | 29 | 1 | I/O | ST | RC7 can also be the USART Asynchronous Receive or Synchronous Data. |
| | | | | | | PORTD is a bi-directional I/O port or parallel slave port when interfacing to a microprocessor bus. |
| RD0/PSP0 | 19 | 21 | 38 | I/O | ST/TTL(3) | |
| RD1/PSP1 | 20 | 22 | 39 | I/O | ST/TTL(3) | |
| RD2/PSP2 | 21 | 23 | 40 | I/O | ST/TTL(3) | |
| RD3/PSP3 | 22 | 24 | 41 | I/O | ST/TTL(3) | |
| RD4/PSP4 | 27 | 30 | 2 | I/O | ST/TTL(3) | |
| RD5/PSP5 | 28 | 31 | 3 | I/O | ST/TTL(3) | |
| RD6/PSP6 | 29 | 32 | 4 | I/O | ST/TTL(3) | |
| RD7/PSP7 | 30 | 33 | 5 | I/O | ST/TTL(3) | |
| | | | | | | PORTE is a bi-directional I/O port. |
| RE0/$\overline{RD}$/AN5 | 8 | 9 | 25 | I/O | ST/TTL(3) | RE0 can also be read control for the parallel slave port, or analog input5. |
| RE1/$\overline{WR}$/AN6 | 9 | 10 | 26 | I/O | ST/TTL(3) | RE1 can also be write control for the parallel slave port, or analog input6. |
| RE2/$\overline{CS}$/AN7 | 10 | 11 | 27 | I/O | ST/TTL(3) | RE2 can also be select control for the parallel slave port, or analog input7. |
| VSS | 12,31 | 13,34 | 6,29 | P | — | Ground reference for logic and I/O pins. |
| VDD | 11,32 | 12,35 | 7,28 | P | — | Positive supply for logic and I/O pins. |
| NC | — | 1,17,28, 40 | 12,13, 33,34 | | — | These pins are not internally connected. These pins should be left unconnected. |

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

**Note 1:** This buffer is a Schmitt Trigger input when configured as an external interrupt.
**2:** This buffer is a Schmitt Trigger input when used in Serial Programming mode.
**3:** This buffer is a Schmitt Trigger input when configured as general purpose I/O and a TTL input when used in the Parallel Slave Port mode (for interfacing to a microprocessor bus).
**4:** This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

19-0065; Rev 6; 10/03

MAXIM

# +5V, RS-232 Transceivers with 0.1µF External Capacitors



## General Description

The MAX200–MAX211/MAX213 transceivers are designed for RS-232 and V.28 communication interfaces where ±12V supplies are not available. On-board charge pumps convert the +5V input to the ±10V needed for RS-232 output levels. The MAX201 and MAX209 operate from +5V and +12V, and contain a +12V to -12V charge-pump voltage converter.

The MAX200–MAX211/MAX213 drivers and receivers meet all EIA/TIA-232E and CCITT V.28 specifications at a data rate of 20kbps. The drivers maintain the ±5V EIA/TIA-232E output signal levels at data rates in excess of 120kbps when loaded in accordance with the EIA/TIA-232E specification.

The 5µW shutdown mode of the MAX200, MAX205, MAX206, and MAX211 conserves energy in battery-powered systems. The MAX213 has an active-low shutdown and an active-high receiver enable control. Two receivers of the MAX213 are active, allowing ring indicator (RI) to be monitored easily using only 75µW power.

The MAX211 and MAX213 are available in a 28-pin wide small-outline (SO) package and a 28-pin shrink small-outline (SSOP) package, which occupies only 40% of the area of the SO. The MAX207 is now available in a 24-pin SO package and a 24-pin SSOP. The MAX203 and MAX205 use no external components, and are recommended for applications with limited circuit board space.

## Applications

Computers
  Laptops, Palmtops, Notebooks

Battery-Powered Equipment

Hand-Held Equipment

## Next-Generation Device Features

- **For Low-Cost Applications:**
  **MAX221E: ±15kV ESD-Protected, +5V, 1µA, Single RS-232 Transceiver with AutoShutdown™**
- **For Low-Voltage and Space-Constrained Applications:**
  **MAX3222E/MAX3232E/MAX3237E/MAX3241E/MAX3246E: ±15kV ESD-Protected, Down to 10nA, +3.0V to +5.5V, Up to 1Mbps, True RS-232 Transceivers (MAX3246E Available in UCSP™ Package)**
- **For Space-Constrained Applications:**
  **MAX3228E/MAX3229E: ±15kV ESD-Protected, +2.5V to +5.5V, RS-232 Transceivers in UCSP**
- **For Low-Voltage or Data Cable Applications:**
  **MAX3380E/MAX3381E: +2.35V TO +5.5V, 1µA, 2Tx/2Rx RS-232 Transceivers with ±15kV ESD-Protected I/O and Logic Pins**
- **For Low-Power Applications:**
  **MAX3224E–MAX3227E/MAX3244E/MAX3245E: ±15kV ESD-Protected, 1µA, 1Mbps, +3.0V to +5.5V, RS-232 Transceivers with AutoShutdown Plus™**

*Ordering Information appears at end of data sheet*



## Selector Guide

| Part Number | Power-Supply Voltage (V) | No. of RS-232 Drivers | No. of RS-232 Receivers | No. of Receivers Active in Shutdown | No. of External Capacitors (0.1µF) | Low-Power Shutdown/ TTL Three-State |
|---|---|---|---|---|---|---|
| MAX200 | +5 | 5 | 0 | 0 | 4 | Yes/No |
| MAX201 | +5 and +9.0 to +13.2 | 2 | 2 | 0 | 2 | No/No |
| MAX202 | +5 | 2 | 2 | 0 | 4 | No/No |
| MAX203 | +5 | 2 | 2 | 0 | None | No/No |
| MAX204 | +5 | 4 | 0 | 0 | 4 | No/No |
| MAX205 | +5 | 5 | 5 | 0 | None | Yes/Yes |
| MAX206 | +5 | 4 | 3 | 0 | 4 | Yes/Yes |
| MAX207 | +5 | 5 | 3 | 0 | 4 | No/No |
| MAX208 | +5 | 4 | 4 | 0 | 4 | No/No |
| MAX209 | +5 and +9.0 to +13.2 | 3 | 5 | 0 | 2 | No/Yes |
| MAX211 | +5 | 4 | 5 | 0 | 4 | Yes/Yes |
| MAX213 | +5 | 4 | 5 | 2 | 4 | Yes/Yes |

# +5V RS-232 Transceivers with 0.1μF External Capacitors

## ABSOLUTE MAXIMUM RATINGS

$V_{CC}$ .......... -0.3V to +6V
V+ .......... ($V_{CC}$ - 0.3V) to +14V
V- .......... +0.3V to -14V
Input Voltages
$T_{IN}$ .......... -0.3V to ($V_{CC}$ + 0.3V)
$R_{IN}$ .......... ±30V
Output Voltages
$T_{OUT}$ .......... (V+ + 0.3V) to (V- - 0.3V)
$R_{OUT}$ .......... -0.3V to ($V_{CC}$ + 0.3V)
Short-Circuit Duration
$T_{OUT}$ .......... Continuous
Continuous Power Dissipation ($T_A$ = +70°C)
14-Pin Plastic DIP (derate 10.00mW/°C above +70°C) .800mW
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C) 842mW
16-Pin SO (derate 8.70mW/°C above +70°C) .......... 696mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C) .. 762mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C) .. 800mW
20-Pin Plastic DIP (derate 11.11mW/°C above +70°C) 889mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C) .. 800mW
20-Pin CERDIP (derate 11.11mW/°C above +70°C) .. 889mW
24-Pin Narrow Plastic DIP (derate 13.33mW/°C above +70°C) 1067mW
24-Pin Wide Plastic DIP (derate 9.09mW/°C above+70°C) 727mW
24-Pin Wide SO (derate 11.76mW/°C above +70°C) .. 941mW
24-Pin SSOP (derate 8.00mW/°C above +70°C) .......... 640mW
24-Pin CERDIP (derate 12.50mW/°C above +70°C) . 1000mW
28-Pin Wide SO (derate 12.50mW/°C above +70°C) . 1000mW
28-Pin SSOP (derate 9.52mW/°C above +70°C) .......... 762mW
Operating Temperature Ranges:
MAX2_ _C_ _ .......... 0°C to +70°C
MAX2_ _E_ _ .......... -40°C to +85°C
MAX2_ _M_ _ .......... -55°C to +125°C
Storage Temperature Range .......... -65°C to +160°C
Lead Temperature (soldering, 10sec) .......... +300°C

*Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

## ELECTRICAL CHARACTERISTICS

(MAX202/204/206/208/211/213 $V_{CC}$ = 5V ±10%, MAX200/203/205/207 $V_{CC}$ = 5V ±5%, C1–C4 = 0.1μF, MAX201/MAX209 $V_{CC}$ = 5V ±10%, V+ = 9.0V to 13.2V, $T_A$ = $T_{MIN}$ to $T_{MAX}$, unless otherwise noted.)

| PARAMETER | CONDITIONS | | MIN | TYP | MAX | UNITS |
|---|---|---|---|---|---|---|
| Output Voltage Swing | All transmitter outputs loaded with 3kΩ to ground | | ±5 | ±8 | | V |
| $V_{CC}$ Power-Supply Current | No load, $T_A$ = +25°C | MAX202, MAX203 | | 8 | 15 | mA |
| | | MAX200, MAX204-MAX208, MAX211, MAX213 | | 11 | 20 | |
| | | MAX201, MAX209 | | 0.4 | 1 | |
| V+ Power-Supply Current | No load | MAX201 | | 5 | 10 | mA |
| | | MAX209 | | 7 | 15 | |
| Shutdown Supply Current | Figure 1, $T_A$ = +25°C | MAX200, MAX205, MAX206, MAX211 | | 1 | 10 | μA |
| | | MAX213 | | 15 | 50 | |
| Input Logic Threshold Low | $T_{IN}$, $\overline{EN}$, SHDN, EN, $\overline{SHDN}$ | | | | 0.8 | V |
| Input Logic Threshold High | $T_{IN}$ | | 2.0 | | | V |
| | $\overline{EN}$, SHDN, EN, $\overline{SHDN}$ | | 2.4 | | | |
| Logic Pull-Up Current | $T_{IN}$ = 0V | | | 15 | 200 | μA |
| RS-232 Input Voltage Operating Range | | | -30 | | +30 | V |
| Receiver Input Threshold Low | $V_{CC}$ = 5V, $T_A$ = +25°C | Active mode | 0.8 | 1.2 | | V |
| | | Shutdown mode, MAX213, R4, R5 | 0.6 | 1.5 | | |
| Receiver Input Threshold High | $V_{CC}$ = 5V, $T_A$ = +25°C | Active mode | | 1.7 | 2.4 | V |
| | | Shutdown mode, MAX213, R4, R5 | | 1.5 | 2.4 | |
| RS-232 Input Hysteresis | $V_{CC}$ = 5V, no hysteresis in shutdown | | 0.2 | 0.5 | 1.0 | V |
| RS-232 Input Resistance | $V_{CC}$ = 5V, $T_A$ = +25°C | | 3 | 5 | 7 | kΩ |

## ELECTRICAL CHARACTERISTICS (continued)

(MAX202/204/206/208/211/213 $V_{CC}$ = 5V ±10%, MAX200/203/205/207 $V_{CC}$ = 5V ±5%, C1-C4 = 0.1μF, MAX201/MAX209 $V_{CC}$ = 5V ±10%, V+ = 9.0V to 13.2V, $T_A$ = $T_{MIN}$ to $T_{MAX}$, unless otherwise noted.)

| PARAMETER | CONDITIONS | | MIN | TYP | MAX | UNITS |
|---|---|---|---|---|---|---|
| TTL/CMOS Output Voltage Low | $I_{OUT}$ = 3.2mA (MAX201, MAX202, MAX203), $I_{OUT}$ = 1.6mA (all others) | | | | 0.4 | V |
| TTL/CMOS Output Voltage High | $I_{OUT}$ = 1.0mA | | 3.5 | | | V |
| TTL/CMOS Output Leakage Current | $\overline{EN}$ = $V_{CC}$, EN = 0V, 0V ≤ $R_{OUT}$ ≤ $V_{CC}$ | | | 0.05 | ±10 | μA |
| Output Enable Time (Figure 2) | MAX205, MAX206, MAX209, MAX211, MAX213 | | | 600 | | ns |
| Output Disable Time (Figure 2) | MAX205, MAX206, MAX209, MAX211, MAX213 | | | 200 | | ns |
| Receiver Propagation Delay | MAX213 | $\overline{SHDN}$ = 0V, R4, R5 | | 4 | 40 | μs |
| | MAX213 | $\overline{SHDN}$ = $V_{CC}$ | | 0.5 | 10 | μs |
| | MAX200-MAX211 | | | 0.5 | 10 | μs |
| Transmitter Output Resistance | $V_{CC}$ = V+ = V- = 0V, $V_{OUT}$ = ±2V | | 300 | | | Ω |
| Transition Region Slew Rate | $C_L$ = 50pF to 2500pF, $R_L$ = 3kΩ to 7kΩ, $V_{CC}$ = 5V, $T_A$ = +25°C measured from +3V to -3V or -3V to +3V | MAX200, MAX202-MAX211, MAX213 | 3 | 5.5 | 30 | V/μs |
| | | MAX201 | | 4 | 30 | V/μs |
| RS-232 Output Short-Circuit Current | | | | ±10 | ±60 | mA |
| Maximum Data Rate | $R_L$ = 3kΩ to 7kΩ, $C_L$ = 50pF to 1000pF, one transmitter | | 120 | | | kbps |

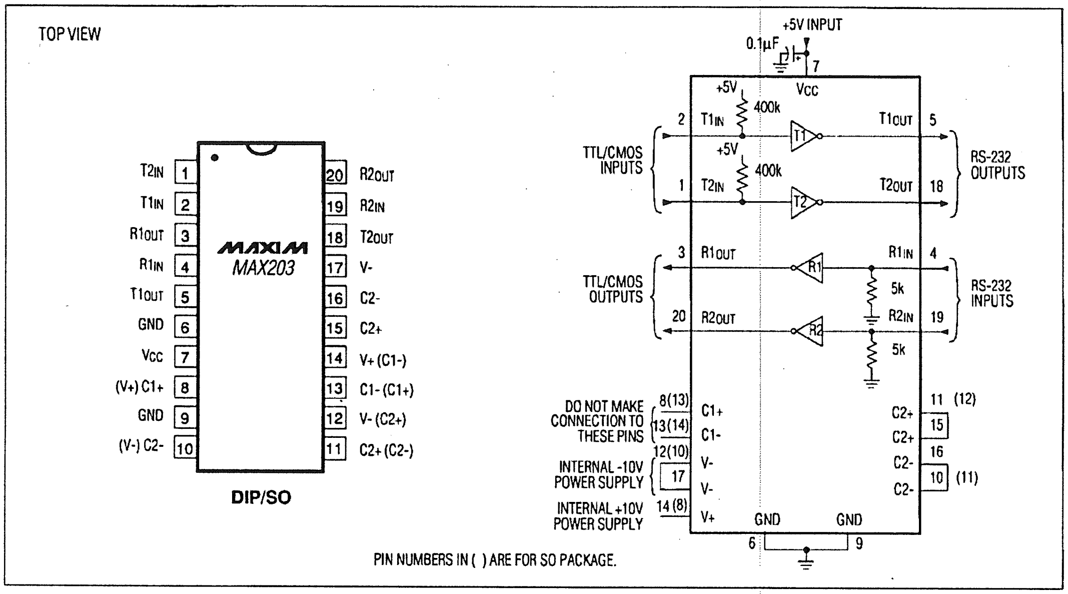



*Figure 9. MAX203 Pin Configuration and Typical Operating Circuit*

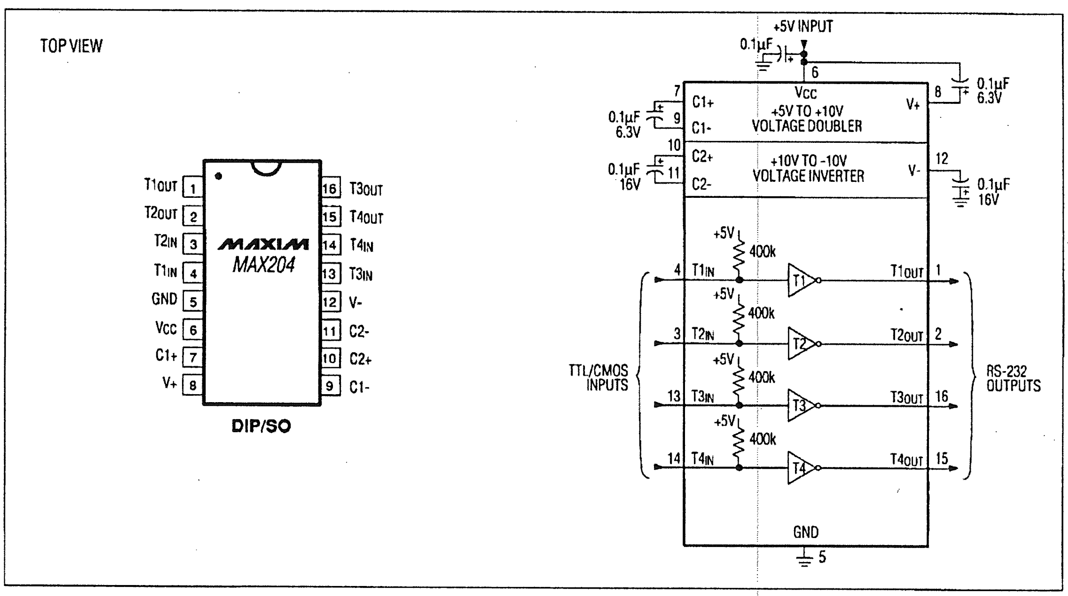



*Figure 10. MAX204 Pin Configuration and Typical Operating Circuit*

19-0481; Rev 2; 11/92

MAXIM

# General Purpose Timers



## General Description

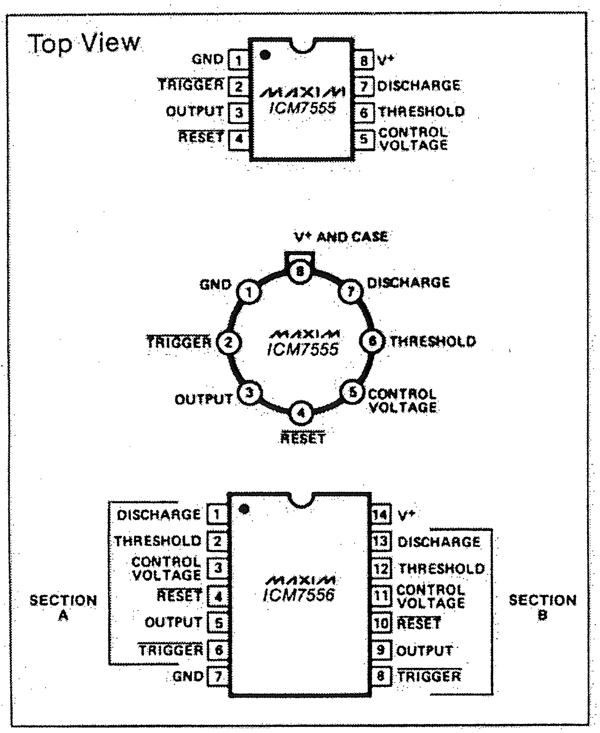
The Maxim ICM7555 and ICM7556 are respectively single and dual general purpose RC timers capable of generating accurate time delays or frequencies. The primary feature is an extremely low supply current, making this device ideal for battery-powered systems. Additional features include low THRESHOLD, $\overline{\text{TRIGGER}}$, and $\overline{\text{RESET}}$ currents, a wide operating supply voltage range, and improved performance at high frequencies.

These CMOS low-power devices offer significant performance advantages over the standard 555 and 556 bipolar timers. Low-power consumption, combined with the virtually non-existent current spike during output transitions, make these timers the optimal solution in many applications.

## Applications

| | |
|---|---|
| Pulse Generator | Pulse Position Modulation |
| Precision Timing | Sequential Timing |
| Time Delay Generation | Missing Pulse Detector |
| Pulse Width Modulation | |

## Pin Configuration



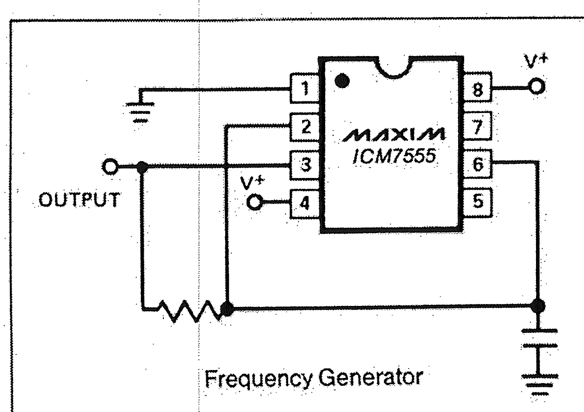

## Features

- ♦ Improved 2nd Source! (See 3rd page for "Maxim Advantage™").
- ♦ Wide Supply Voltage Range: 2-18V
- ♦ No Crowbarring of Supply During Output Transition
- ♦ Adjustable Duty Cycle
- ♦ Low THRESHOLD, TRIGGER and RESET Curents
- ♦ TTL Compatible
- ♦ Monolithic, Low Power CMOS Design

## Ordering Information

| PART | TEMP. RANGE | PACKAGE |
|---|---|---|
| ICM7555IPA | -20°C to +85°C | 8 Lead Plastic DIP |
| ICM7555IJA | -20°C to +85°C | 8 Lead CERDIP |
| ICM7555ITV | -20°C to +85°C | TO-99 Can |
| ICM7555MJA | -55°C to +125°C | 8 Lead CERDIP |
| ICM7555MTV | -55°C to +125°C | TO-99 Can |
| ICM7555ISA | -20°C to +85°C | 8 Lead Small Outline |
| ICM7555/D | 0°C to +70°C | Dice |
| ICM7556IPD | -20°C to +85°C | 14 Lead Plastic DIP |
| ICM7556MJD | -55°C to +125°C | 14 Lead CERDIP |
| ICM7556ISD | -20°C to +85°C | 14 Lead Small Outline |
| ICM7556/D | 0°C to +70°C | Dice |

## Typical Operating Circuit




(Detailed Circuit Diagram — Figure-1a)

# General Purpose Timers

## ABSOLUTE MAXIMUM RATINGS (Note 1)

Supply Voltage ........................................ +18 Volts
Input Voltage TRIGGER
Control Voltage THRESHOLD $<V^+ + 0.3V$ to $\geq -0.3V$
RESET
Output Current ........................................ 100mA
Power Dissipation[2] ICM7556 ........................ 300mW
ICM7555 ........................ 200mW
Operating Temperature Range
ICM7555IJA (Maxim) .................. −20°C to +85°C
ICM7555ISA (Maxim) .................. −20°C to +85°C
ICM7555IPA .................. −20°C to +85°C
ICM7555ITV .................. −20°C to +85°C
ICM7556IPD .................. −20°C to +85°C
ICM7555MTV .................. −55°C to +125°C
ICM7556MJD .................. −55°C to +125°C
Storage Temperature .................. −65°C to +150°C
Lead Temperature (Soldering 60 Seconds) .................. +300°C

Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions above those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## ELECTRICAL CHARACTERISTICS

($V^+ = +2$ to $+15$ volts; $T_A = 25°C$, Unless Noted)

| PARAMETER | SYMBOL | TEST CONDITIONS | VALUE MIN | TYP | MAX | UNITS |
|---|---|---|---|---|---|---|
| Supply Voltage | $V^+$ | $-20°C \leq T_A \leq +70°C$ | 2 | | 18 | V |
| | | $-55°C \leq T_A \leq +125°C$ | 3 | | 16 | V |
| Supply Current[3] | $I^+$ | ICM7555 $V^+ = 2V$ | | 60 | 200 | µA |
| | | $V^+ = 18V$ | | 120 | 300 | µA |
| | | ICM7556 $V^+ = 2V$ | | 120 | 400 | µA |
| | | $V^+ = 18V$ | | 240 | 600 | µA |
| Timing Error | | $R_A$, $R_B$ = 1k to 100k, $5V \leq V^+ \leq 15V$; C = 0.1µF | | | | |
| Initial Accuracy | | Note 4 | | 2.0 | 5.0 | % |
| Drift with Temperature | | Note 4 $V^+ = 5V$ | | 50 | | ppm/°C |
| | | $V^+ = 10V$ | | 75 | | |
| | | $V^+ = 15V$ | | 100 | | |
| Drift with Supply Voltage | | $V^+ = 5V$ | | 1.0 | 3.0 | %/V |
| Threshold Voltage | $V_{TH}$ | $V^+ = 5V$ | 0.63 | 0.66 | 0.67 | $V^+$ |
| Trigger Voltage | $V_{TRIG}$ | $V^+ = 5V$ | 0.29 | 0.33 | 0.34 | $V^+$ |
| Trigger Current | $I_{TRIG}$ | $V^+ = 18V$ | | 50 | | pA |
| | | $V^+ = 5V$ | | 10 | | pA |
| | | $V^+ = 2V$ | | 1 | | pA |
| Threshold Current | $I_{TH}$ | $V^+ = 18V$ | | 50 | | pA |
| | | $V^+ = 5V$ | | 10 | | pA |
| | | $V^+ = 2V$ | | 1 | | pA |
| Reset Current | $I_{RST}$ | $V_{RESET}$ = Ground $V^+ = 18V$ | | 100 | | pA |
| | | $V^+ = 5V$ | | 20 | | pA |
| | | $V^+ = 2V$ | | 2 | | pA |
| Reset Voltage | $V_{RST}$ | $V^+ = 18V$ | 0.4 | 0.7 | 1.0 | V |
| | | $V^+ = 2V$ | 0.4 | 0.7 | 1.0 | V |
| Control Voltage Lead | $V_{CV}$ | $V^+ = 5V$ | 0.62 | 0.66 | 0.67 | $V^+$ |
| Output Voltage Drop | $V_O$ | Output Lo $V^+ = 18V$ $I_{SINK} = 3.2mA$ | | 0.1 | 0.4 | V |
| | | $V^+ = 5V$ $I_{SINK} = 3.2mA$ | | 0.15 | 0.4 | V |
| | | Output Hi $V^+ = 18V$ $I_{SOURCE} = 1.0mA$ | 17.25 | 17.8 | | V |
| | | $V^+ = 5V$ $I_{SOURCE} = 1.0mA$ | 4.0 | 4.5 | | V |
| Rise Time of Output | $t_r$ | $R_L = 10M\Omega$ $C_L = 10pF$ $V^+ = 5V$ | 35 | 40 | 75 | ns |
| Fall Time of Output | $t_f$ | $R_L = 10M\Omega$ $C_L = 10pF$ $V^+ = 5V$ | 35 | 40 | 75 | ns |
| Guaranteed Max Osc Freq | $f_{max}$ | Astable Operation | 500 | | | kHz |



**Note 1:** Due to the SCR structure inherent in the CMOS process used to fabricate these devices, connecting any terminal to a voltage greater than $V^+ + 0.3V$ or less than $V^- - 0.3V$ may cause destructive latchup. For this reason it is recommended that no inputs from external sources not operating from the same power supply be applied to the device before its power supply is established. In multiple systems, the supply of the ICM7555/6 must be turned on first.

**Note 2:** Junction temperatures should not exceed 135°C and the power dissipation must be limited to 20mW at 125°C. Below 125°C power dissipation may be increased to 300mW at 25°C. Derating factor is approximately 3mW/°C (7556) or 2mW/°C (7555).

**Note 3:** The supply current value is essentially independent of the TRIGGER, THRESHOLD and RESET voltages.

**Note 4:** Parameter is not 100% tested. Majority of all units meet this specification.

*The electrical characteristics above are a reproduction of a portion of Intersil's copyrighted (1983/1984) data book. This information does not constitute any representation by Maxim that Intersil's products will perform in accordance with these specifications. The "Electrical Characteristics Table" along with the descriptive excerpts from the original manufacturer's data sheet have been included in this data sheet solely for comparative purposes.*

/V\/\X/\/V\
# ADVANTAGE™
## General Purpose Timers

ICM7555/7556

- ♦ Lower Supply Current
- ♦ Increased Output Source Current
- ♦ Guaranteed THRESHOLD, $\overline{\text{TRIGGER}}$ and $\overline{\text{RESET}}$ Input Currents
- ♦ Guaranteed Discharge Output Voltage
- ♦ Supply Current Guaranteed Over Temperature
- ♦ Significantly Improved ESD Protection (Note 6)
- ♦ Maxim Quality and Reliability

## ABSOLUTE MAXIMUM RATINGS
This device conforms to the Absolute Maximum Ratings on adjacent page.

## ELECTRICAL CHARACTERISTICS
Specifications below satisfy or exceed all "tested" parameters on adjacent page.
($V^+ = +2$ to $+15$ volts; $T_A = 25°C$, unless noted.)

| PARAMETER | SYMBOL | TEST CONDITIONS | MIN | TYP | MAX | UNITS |
|---|---|---|---|---|---|---|
| Supply Voltage | $V^+$ | $-20°C \leq T_A \leq +85°C$ | 2 | | 16.5 | V |
| | | $-55°C \leq T_A \leq +125°C$ | 3 | | 16 | V |
| **Supply Current** (Note 3) | $I^+$ | **ICM 7555** $V^+ = 2\text{-}16.5V$; $T_A = +25°C$ | | **30** | **250** | µA |
| | | $V^+ = 5V$; $T_A = +25°C$ | | | **120** | µA |
| | | $V^+ = 5V$; $-20°C \leq T_A \leq +85°C$ | | | **250** | µA |
| | | $V^+ = 5V$; $-55°C \leq T_A \leq +125°C$ | | | **300** | µA |
| | | **ICM 7556** $V^+ = 2\text{-}16.5V$; $T_A = +25°C$ | | **60** | **500** | µA |
| | | $V^+ = 5V$; $T_A = +25°C$ | | | **240** | µA |
| | | $V^+ = 5V$; $-20°C \leq T_A \leq +85°C$ | | | **500** | µA |
| | | $V^+ = 5V$; $-55°C \leq T_A \leq +125°C$ | | | **600** | µA |
| Timing Error (Note 4) | | Circuit of figure 1(b); $R_A = R_B = 100k\Omega$, $C = 0.1\mu F$, $V^+ = 5V$ | | | | |
| Initial Accuracy (Note 5) | | | | 2.0 | 5.0 | % |
| Drift with Temperature | | $V^+ = 5V$ | | 50 | | ppm/°C |
| | | $V^+ = 10V$ | | 75 | | ppm/°C |
| | | $V^+ = 15V$ | | 100 | | ppm/°C |
| Drift with Supply Voltage | | $V^+ = 5V$ | | 1.0 | 3.0 | %/V |
| Threshold Voltage | $V_{TH}$ | $V^+ = 5V$ | 0.63 | 0.66 | 0.67 | $V^+$ |
| Trigger Voltage | $V_{TRIG}$ | $V^+ = 5V$ | 0.29 | 0.33 | 0.34 | $V^+$ |
| **Trigger Current** | $I_{TRIG}$ | $V^+ = 16.5V$ | | **50** | | pA |
| | | $V^+ = 5V$ | | 10 | | pA |
| | | $V^+ = 2V$ | | 1 | | pA |
| **Threshold Current** | $I_{TH}$ | $V^+ = 16.5V$ | | **50** | | pA |
| | | $V^+ = 5V$ | | 10 | | pA |
| | | $V^+ = 2V$ | | 1 | | pA |
| **Reset Current** | $I_{RST}$ | $V_{RESET}$ = Ground $V^+ = 16.5V$ | | **100** | | pA |
| | | $V^+ = 5V$ | | 20 | | pA |
| | | $V^+ = 2V$ | | 2 | | pA |
| Reset Voltage | $V_{RST}$ | $V^+ = 16.5V$ | 0.4 | 0.7 | 1.2 | V |
| | | $V^+ = 2V$ | 0.4 | 0.7 | 1.2 | V |
| Control Voltage | $V_{CV}$ | $V^+ = 5V$ | 0.62 | 0.66 | 0.67 | $V^+$ |
| **Output Voltage Drop** | $V_O$ | Output Lo $V^+ = 16.5V$ $I_{SINK} = 3.2mA$ | | 0.1 | 0.4 | V |
| | | $V^+ = 5V$ $I_{SINK} = 3.2mA$ | | 0.15 | 0.4 | V |
| | | **Output Hi** $V^+ = 16.5V$ $I_{SOURCE} = 2.0mA$ | 15.75 | 16.25 | | V |
| | | $V^+ = 5V$ $I_{SOURCE} = 2.0mA$ | 4.0 | 4.5 | | V |
| **Discharge Output Voltage** | $V_{DIS}$ | $V^+ = 5V$, $I_{DIS} = 3.2mA$ | | **0.1** | **0.4** | **V** |
| Rise Time of Output (Note 4) | $t_r$ | $R_L = 10M\Omega$ $C_L = 10pF$ $V^+ = 5V$ | 35 | 40 | 75 | ns |
| Fall Time of Output (Note 4) | $t_f$ | $R_L = 10M\Omega$ $C_L = 10pF$ $V^+ = 5V$ | 35 | 40 | 75 | ns |
| Guaranteed Max Osc. Freq. (Note 4) | $f_{max}$ | Astable Operation | 500 | | | kHz |

Note 1: Due to the SCR structure inherent in the CMOS process used to fabricate these devices, connecting any terminal to a voltage greater than $V^+ +0.3V$ or less than $V^- -0.3V$ may cause destructive latchup. For this reason it is recommended that no inputs from external sources not operating from the same power supply be applied to the device before its power supply is established. In multiple systems, the supply of the ICM7555/6 must be turned on first.

Note 2: Junction temperatures should not exceed 135°C and the power dissipation must be limited to 20mW at 125°C. Below 125°C power dissipation may be increased to 300mW at 25°C. Derating factor is approximately 3mW/°C (7556) or 2mW/°C (7555).

Note 3: The supply current value is essentially independent of the $\overline{\text{TRIGGER}}$, THRESHOLD AND $\overline{\text{RESET}}$ voltages.

Note 4: Parameter is not 100% tested. Majority of all units meet this specification.

Note 5: Deviation from $f = 1.46/(R_A + 2R_B)C$, $V^+ = 5V$.

Note 6: All pins are designed to withstand electrostatic discharge (ESD) levels in excess of 2000V. (Mil Std 883B, Method 3015.1 Test Circuit.)