The MiniTest 488 Test Controller:
When smaller and simpler just fits better.

The trend in electronics toward faster microprocessors and larger memories has become as inevitable as the rising sun and Microsoft's growing earnings. At the ISSCC conference in February 1997, 4 Gbit DRAMs and 600-MHz microprocessors were presented. Another, albeit less obvious, trend is for the ever-decreasing scale of small systems. For example, this past year, the first 8 pin RISC microcontroller was introduced by Microchip of Chandler AZ. Dallas Semiconductor is supplying memories in three pin packages. These miniaturized and scaled-back approaches provide efficiency and cost benefits in many systems and often have wider application than the "Mega devices".

Figure 1, MT488A

The scale of the behemoth memories and swift microprocessors being developed at the top end of the spectrum is not suitable for a lamp dimmer or an infrared controller like the Microchip (PIC12C508) eight pin RISC processor. And in the same way, using a Pentium class PC and an ISA IEEE 488 card as a test controller is not always the most efficient approach for all applications.

This article introduces a new portable test controller called the MT488 (Mini-Test) and describes its features and its applications. A specific automatic test implementation at RF component manufacturer G.T Microwave will be described. The device was developed by Tidal Engineering and is being marketed by ADI American Distributors, Inc. both from Whippany, NJ.

Test Application

G.T. Microwave is one of the foremost suppliers of digitally controlled microwave RF attenuators and vector modulators. These devices use state of the art RF design techniques and offer accuracy and speed that set the standard for the industry.

Please refer to the article titled "A Broadband I & Q Vector Modulator" in the December 1996 issue of the Microwave Journal for a detailed description of this technology. These devices use EEPROM calibration tables to linearize the PIN diode response of the attenuators and to provide precision that would be impossible otherwise. During attenuator calibration, the device is connected to an RF network analyzer. Adjustments are made to the unit under test during this calibration using the data from the network analyzer.

Originally G.T. Microwave built a small test system using a PC, a National Instruments(tm) IEEE 488 interface card and a digital I/O card. Configuring the hardware and software was time-consuming because of all the different software layers. These include DOS/Windows peripheral drivers and language drivers for each device.

Tidal Engineering and G.T. Microwave began talking about the possibility of using ADI's new SBC488-OEM as a rapid development, portable test controller. G.T. Microwave was interested in adding another production test and calibration station and was also interested in developing a demonstration controller to be used at trade shows. The SBC488-OEM is an IEEE 488 interface being marketed to power supply and test equipment OEM's as a flexible rapid prototyping IEEE 488/RS-232/RS-485 interface and had all the features required for the application. Work began porting G.T.Microwave's existing basic software to the SBC488 in early January and after ten hours the test program was up and running in C on the MT488.

MiniTest 488 Solution

The MT488 system offers many advantages for this specific test application. There are no jumpers to configure or Dos/Windows/Drivers to load. Just apply power and start to program. Once programmed, there is no boot-up delay, the MT488 is "Instant On". The 2 lb. unit is easily carried to a trade show and around the lab.

Homogeneous C Language Environment

The traditional PC environment is a wonderful example of the way standards allow us to build systems from software and hardware components from many suppliers. Unfortunately, it is not always easy to get different components to work together. Setting up interrupts, I/O addresses, dip switches and finding conflicts can sometimes be arduous even for the initiated. The MT488 uses a fully integrated Windows(tm) based C development environment. During development, program download and debug information is sent from the development PC to the serial port of the MT488. Once debugged, a program can be run from non-volatile SRAM or burned into EPROM. The environment provides editing, compiling, linking and debugging capabilities with just a PC. The language features Real-Time multi-tasking, floating point and a robust library. The following are some of its debugging capabilities and library functions:

1. Single stepping into and around function calls.
2. Break Points.
3. Symbolic and assembly language debugging.
4. Stdio for debug and data input and output.
5. Watch Expressions.

The libraries include:

1. String libraries
2. OPTO-22® RS-485 networking software.
3. Math including floating point.
4. IEEE 488 device drivers.
5. Peripheral drivers.

Low system cost

The hardware cost of the PC and MT488 based systems used by G.T. Microwave are compared as follows:

IBM PC Compatible	$1,500
Nat. Inst. GPIB-PCII	$395
PC-DIO96	$295
Total	**$2,190**

MT488 System	$575
LCD and Keypad	$165
Digital I/O (32 Bits)	$ 35
Total	**$775**

Software costs for these two approaches are the same, roughly $400 for development tools.

Software Compatibility

The MT488's IEEE 488 drivers are compatible with National Instruments (tm) C libraries.
Software running on a National Instruments card is easily ported to the MT488 system. Conversely, code written for the MT488 is readily ported to a PC based test system using the National Instruments drivers. This allows flexibility with future requirements and software reusability.

Built in peripherals; digital & analog I/O

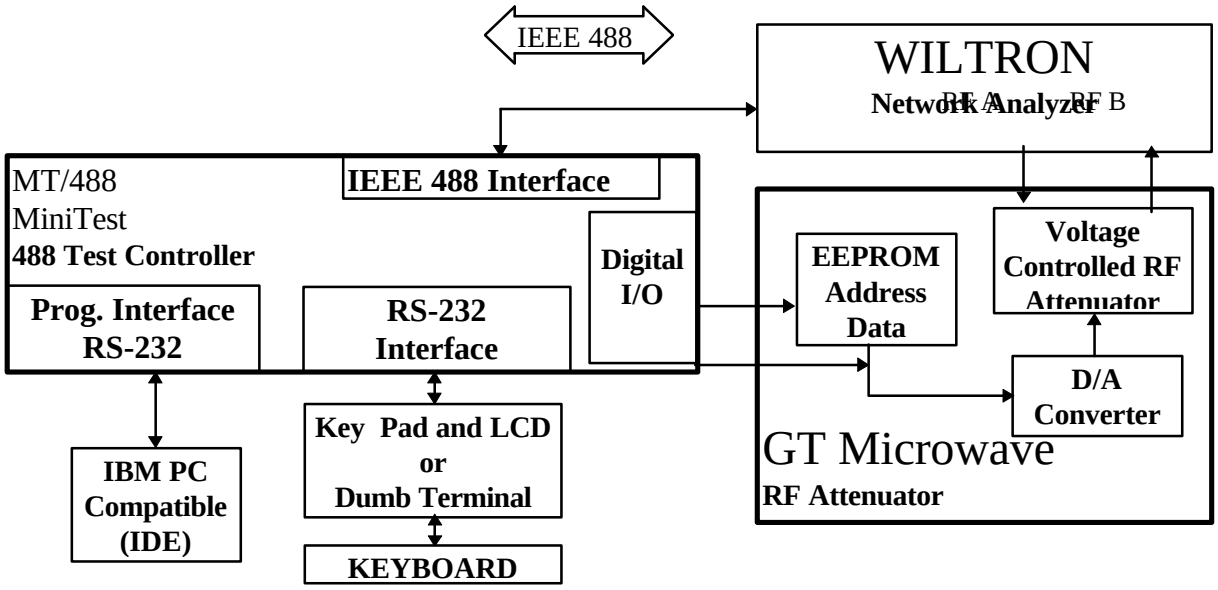
The MT488's built-in peripherals can eliminate the need for certain IEEE 488 instruments and ISA PC devices. In this application, G.T. Microwave eliminated the separate digital I/O card and replaced it with the built-in digital I/O of the MT488.

Conclusion

The MT488 is a small yet capable test controller, useful when a large system is not necessary or desirable. It's cost benefits and advantages including ease of use are obvious. For more information contact:

ADI American Distributors, Inc.
2 Emery Avenue, Randolph 07869
Ask for Craig Borax 973-328-1181

G.T. Microwave, Inc.
10 Orben Drive, Landing, NJ 07850
Ask for Tony Baliotis 973-770-7300



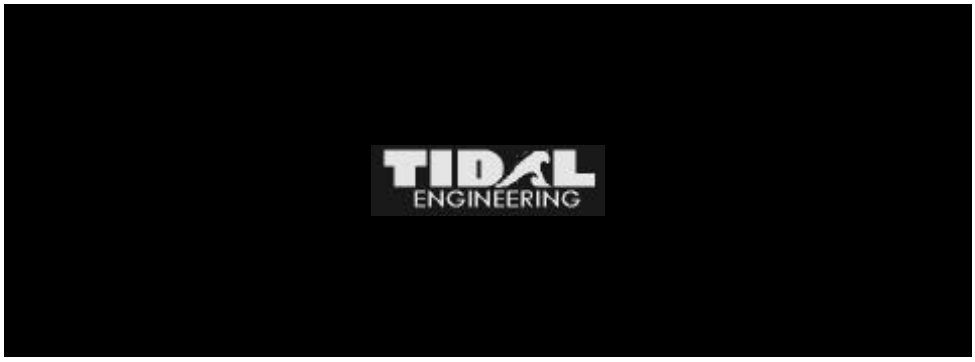
Software

```
main() {
     int  pattern_in[MAX_PORT];    // input patterns
     int i;
     long lMproCount;
     long lin_bit,lmax_in_count,lInb,lIna,lPatternOut,lMaxProCount,lHproCount;
     long lin_count, lpro_count,lProb,lProa,lStart,lProBit,lLproCount;
     float fmax_attn,fRef,lgoal,fLosMin,fAttn,fLattn,fLosMax,fFlt,flsb;
     float fXl,fXh,fXm,fT4,fGoal,fLatten,fHattn,fMattn;
     int  count;
     char stbuf[TBUFSIZE];      // transmit buffer
     char srbuf[RBUFSIZE];      // receive buffer
     char buf[RBUFSIZE+1];  // dummy buffer for receiving a complete command
     char s, t,data;
     char tbuf[ieee_out_len], rbuf[ieee_in_len];
     int commands,osc;
     unsigned int chan;
     int ii,n,j,z,v,PAD_;
     char temp[25];
     VdInit();                  // initialize the virtual driver
     hitwd();
     init_io();
     fLosMin=1.234E-3;
     ftoa(fLosMin,ieee_in);
     strcpy(ieee_in,menu);
     fLosMin=atof(floatprec());
     da[0]=4095;                    //Initialize the Program line
     set_da();
     leds=255;
     outport(CS_574,255);                    //turn off all leds
          PAD_=6;
     if (init_488()==-1) {
          sprintf("Could not initialize Natational Instruments IEEE 488 ASIC");
          Dwrite_z0 (buf,strlen(buf));
     }
     if ((osc = ibdev(0, PAD_, 0, T10s, 1, 0)) < 0) {
          sprintf("Could not open OSC_01 device");
          Dwrite_z0 (buf,strlen(buf));
     }
     ibren(osc);           //Put Oscillator in REN
     ibloc(osc);
```

```
     //////********from tst1232.c
     // Since serial port 1 is the default programming port with the SmartCore,
     Dinit_z0(srbuf,stbuf,RBUFSIZE,TBUFSIZE, MODE, IBAUD, NO_MODEM, ECHO );
     toggle_pgm();

//-------------------------------------------------------------------------
//       COUNT AND PROGRAM LOOP
     Dwrite_z0 ("Tidal Engineering\n\r",19 );
     while (Dread_z0(buf,ENTER) == 0 ){};
     lin_bit=atoi(buf);  // wait for string terminated with CR
     lmax_in_count= (pow  (2.0 , lin_bit) ) -1;
     sprintf(buf,"INPUT STEPS= %d \n\r",lmax_in_count);
   Dwrite_z0 (buf,strlen(buf)) ;//PRINT " INPUT STEPS= " ; lmax_in_count
     Dwrite_z0 ("Number of program bits=?\n\r",27 );
     while( Dread_z0(buf,ENTER) == 0 ) {};
     lProBit=atoi(buf);  // wait for string terminated with CR
     lMaxProCount= (pow  (2.0 , lProBit) ) -1;//
     sprintf(buf,"Program steps= %d \n\r",lMaxProCount);
   Dwrite_z0 (buf,strlen(buf)) ;
     Dwrite_z0 ("Maximum attenuation=?\n\r",23 );
     while( Dread_z0(buf,ENTER) == 0 ){};
     fmax_attn=atof(buf);  // wait for string terminated with CR
     flsb=fmax_attn / lmax_in_count;
     sprintf(buf,"LSB = %f \n\r",flsb);
   Dwrite_z0 (buf,strlen(buf) );
     lin_count = 0;
     lpro_count = 0;
     fGoal= 0;
     fRef= fGoal;                              //SET REF. ON WILTRON
     sprintf(bofer, "off %f",fRef);
     ibwrt(osc,bofer,strlen(bofer)+1);
     wr_atten_in(lin_count);
     wr_atten_pro(lpro_count);      //SET PROGRAM COUNT
     toggle_pgm();
////////////////////////////////////////////////////////
//                      START PROGRAM LOOP
////////////////////////////////////////////////////////

     Dwrite_z0 (" PROGRAM COUNT START # = ? \n\r",29 );
     while( Dread_z0(buf,ENTER) == 0 ) {};
     lStart=atof(buf);  // wait for string terminated with CR

     lpro_count = lStart;

     for (lin_count = 0;lin_count<=lmax_in_count;lin_count=+4) {
          if (lin_count > lmax_in_count) lin_count= lmax_in_count;
       sprintf (bofer,"INCOUNT,LSB= %l,%f /n/r" , lin_count , flsb);
          Dwrite_z0 (bofer,strlen(bofer) );
          fGoal = -1 * lin_count * flsb;
          printf ("fGoal= %f", fGoal);
          fRef=  fGoal;                 //SET REF. ON WILTRON
          printf ("fRef= %f" , fRef);
          set_atten_off(fRef);
          wr_atten_in(lin_count);              // SET INPUT CODE
          do {
               wr_atten_pro(lpro_count);      //SET PROGRAM COUNT
               //CALL SENDIFC(BD%)            // READ ATTENUATION
               fAttn=rd_db();
               if (fAttn <= fGoal) {
                    break;
               }
               else {
                    lLproCount = lpro_count;
                    fLattn=fAttn;
                    lpro_count = lpro_count + 2;
               }
          } while (1);
          if  (fAttn< fGoal) {
               fHattn= fAttn;
               lHproCount = lpro_count;
               lpro_count = lpro_count - 1;
               wr_atten_pro(lpro_count);
               //CALL SENDIFC(BD%)            // READ ATTENUATION
               fAttn=rd_db();
                    fMattn= fAttn;
                    lMproCount = lpro_count;
                    fXl=abs(fGoal - fLattn);fXm=abs(fGoal - fAttn);fXh=abs(fGoal - fHattn);
                    printf ("XL,XM,XH = %f,%f,%f", fXl,fXm,fXh);
                    //XL# = ABS( fGoal - LATTN#)
                    //XM# = ABS( fGoal - ATTN#)
                    //XH# = ABS( fGoal - HATTN#)
```

```
                    if (( fXl< fXm) && ( fXl < fXh)) {
                         Dwrite_z0 ("LOW COUNT IS BEST\n\r",19 ); //PRINT "LOW COUNT IS BEST"
                         lpro_count = lLproCount;
                         wr_atten_pro(lpro_count);  //SET PROGRAM COUNT
               toggle_pgm(); //i%=2       //TOGGLE PROGRAM LINE
                    }
                    else if ((fXm < fXl) && (fXm < fXh)) {
                         Dwrite_z0 ("MIDDLE COUNT IS BEST\n\r",22 );
                         lpro_count = lMproCount;
lpro_count(lin_count) ; lMproCount
                         wr_atten_pro(lpro_count);  //SET PROGRAM COUNT
               toggle_pgm(); //i%=2       //TOGGLE PROGRAM LINE
                    }
                    else if ((fXh < fXl) && (fXh < fXm)) {
                         Dwrite_z0 ("HIGH COUNT IS BEST\n\r",22 );
                         lpro_count = lHproCount;
lpro_count(lin_count) ; lHproCount

                         wr_atten_pro(lpro_count);  //SET PROGRAM COUNT
               toggle_pgm(); //i%=2       //TOGGLE PROGRAM LINE
                    }
                    else if (fAttn= fGoal) {
                         Dwrite_z0 ("THIS IS PERFECTION\n\r",20 );
     toggle_pgm(); //i%=2       //TOGGLE PROGRAM LINE
}                   //END IF
     }
     }             //NEXT lin_count
}             //Main
/*SBC488Q     Page 2   1/5/96    10:04 AM*/
```

1	2	3	DEL
4	5	6	<
7	8	9	>
0	.	±-/+	ENT



TIDAL ENGINEEERING

MT488

MINI-TEST

IEEE 488 Controller