ATMD-GP2

TDC-GP2 Evaluation System

Datasheet

16TH FEB 2006

acam - solutions in time

Precision Time Interval Measurement

acam mess·electronic

Index

Limited Warranty

The ATMD measurement system with it's components ATMD-MB, ATMD-PC, ATMD-PCI and AM-GP2 is designed and offered as an evaluation system for the integrated circuit TDC-GP2, offered by acam-messelectronic. The hardware are warranted against defects in materials and workmanship for a period of 12 months from the date of shipment, as evidenced by receipts or other documentation. acam-messelectronic will, at its option, repair or replace equipment that proves to be defective during the warranty period.

The information provided herein is believed to be reliable. However, acam-messelectronic assumes no responsibilities for inaccuracies or omissions. acam-messelectronic assumes no responsibility for the use of this information, and all use of this information shall be entirely to the user's own risk. No patent rights or licenses to any of the circuits described herein are implied or granted to any third party. acam-messelectronic does not authorize or warrant any acam-messelectronic product for use in life support devices and/or systems.

Except as specified herein, acam-messelectronic makes no warranties, express or implied, and specifically disclaims any warranty of merchantability or fitness for a particular purpose. Customer's right to recover damages caused by fault or negligence on the part of acam-messelectronic shall be limited to the amount theretofore paid by the customer. acam-messelectronic will not be liable for damages resulting from loss of data, profits, use of products, or identical or consequential damages, even if advised of the possibility thereof. This limitation of the liability of acam-messelectronic will apply regardless of the form of action. Any action against acam-messelectronic must be brought within one year after the cause of action acruses. acam-messelectronic shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failure caused by the owner's failure to follow acam-messelectronic's installation and operation instructions. Further the warranty does not cover damages due to owner's modification of the product, owner's abuse, misuse, or negligent acts and power failure or surges or other events outside reasonable control.

The products ATMD with its components comply with EMC directive 89/336/EEC, applied standard DIN EN 61326, Equipment for Control and Laboratory (For use in electromagnetically controlled environment). Generic immunity standard part 2 (EN 61000-4-4: 0,5KV, -4-6: 1V), In case of strong electromagnetic disturbances there might be a deviation of the output signal from the specification, but only for the duration of the disturbance.

1. Introduction

1.1 System Overview

The ATMD-GP2 evaluation system consists of a motherboard together with the AM-GP2 plug-in module, mounted in a metal case. It is connected to the ATMD-PCI interface card (mounted in the PC) by a SCSI-type cable (although the bus is ATMD specific and not a PCI type).

Ordering numbers:

ATMD-GP2	MNR 1062	Motherboard with 1 AM-GP2 plug-in module incl. Software, manuals and cables
ATMD-PCI	MNR 478	PCI interface

1.2 Features ATMD

- FIFO on motherboard 1K (can be increased to 32K)
- Power supply through PCI interface
- Op. temperature range –25°C ... +70°C
- Maximum data rate for console applications without data display about 48 kHz, with numerical display about 21 kHz, for ATMD-GP2 LabView application about 1.5 to 5 kHz.

1.3 Features AM-GP2

- Measurement Range 1
  - 2 channels with typ. 50 ps resolution RMS
  - 15 ns pulse-pair resolution with 4-fold multihit capability
  - Four events can be measured arbitrarily against each other
  - Windowing for precise stop enable
- Measurement Range 2
  - 2 channels with typ 50 ps resolution RMS
  - Measurement range 500 ns to 4 ms
  - 500 ns pulse-pair resolution with 3-fold multihit capability
  - Each one of the three events can be assigned to an adjustable measuring window with 10 ns resolution
- Temperature Measurement Unit
  - 2 or 4 sensors
  - PT500 / PT1000 or higher
  - Very high resolution: 16 Bit eff. (0,004°C for platinum sensors)
  - Ultra low current consumption (0,08 µA when measuring every 30 seconds)
- General
  - Trigger to Rising and falling edge
  - Clock calibration unit
  - Fire pulse generator
  - Precise stop enable by windowing

1.4 Hard- and Software Installation

Important! All components of the ATMD-System are sensitive to static electricity. Before installing the interface board, please touch a grounded object such as a metal screw on the computer. Handle the interface board by its edges and be careful not to twist it.

Perform the following steps to install the interface board:

1. 🗁 ni-visa
   ----------NIvisa.msi — Run NI Visa instrument drivers installer

2. lvruntimeeng.msi — Run NI LabView runtime installer for Windows 98
   or
   visa320runtime.exe — Run NI LabView runtime installer for Windows NT/2000/XP

3. Turn off your computer. Keep your computer plugged in so that it remains grounded while you install your interface board. Remove the computers cover. Next , align the interfaces edge connector with an 32-bit PCI expansion slot for ATMD-PCI. Then, push the board down into the slot until the board locks into place. It might be a tight fit, but do not force the board into place. Screw the mounting bracket of the interface board to the back panel rail of the computer, check the installation and replace the cover of the computer.

4. Connect the interface board and the external ATMD motherboard via the enclosed cable (for convenience a standard SCSI-2 cable is used, but it is not a SCSI interface!) and turn on your computer.

5. The operating system will ask for a driver. Select from
🗁 AcamAtmdPCI — Inf-Files for registration of PCI-interface under NT/2000/XP
   ---------- ATMD_PCI_9X.inf — For Windows '98
   ---------- ATMD_PCI_NT5.inf — For Windows NT, 2000, XP

6. 🗁 ATMD_GP2_3_0
   ----------🗁 Installer
   ----------setup.exe — Run installer for ATMD-GP2 software

7. To start the ATMD-GP2 software select START/Programs/ATMD_GP2/ATMD_GP2_v3.0.

If you want to write your own C++ based software install the following files for a free access to the I/O ports:

🗁 Driver
   ---------- instdrv.exe — Copies giveio.sys and windrvr.sys into the System32\drivers folder
   When working with Windows NT/2000/XP first install the necessary drivers executing instdrv.cmd. Open the device manager, select menu item 'Show hidden devices' and select folder 'Non-PNP devices'. There you will find the giveio.sys and windrvr.sys. Select under properties the start option 'automatic'.

🗁 PCI
   ---------- atmd_pci.dll — Copy this file into system folder
   ---------- atmd_pci.lib — "

Furhter files on the CD-ROM are:

🗁 Samples
   ---------- GP2-1hit-MR1-autocal etc. — Visual C++ samples

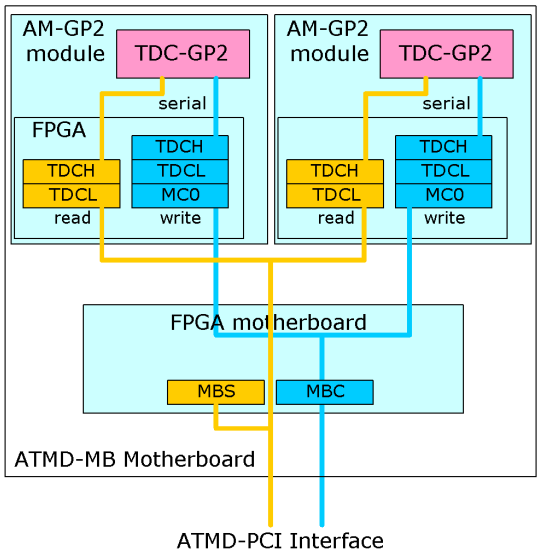
🗁 Configurations
   ---------- MR2.cfg etc. — Sample configurations for the ATMD-GP2 software

🗁 Doc — All available documents in PDF-format

2. Writing Software

2.1 ATMD Registers

The complete control of the measuring module is performed by an FPGA. This FPGA is also responsible for the communication between PC and TDC-GP2. Therefore it provides 4 read/write-registers TDCHx and TDCLx that enables reading and writing to the TDC-GP2. The abbreviation x stands for "0" when communicating with MOD0 (Slot 0) is requested and "1" for read/write access to the AM-GP2 module located in slot 1 (MOD1).



2.1.1 Register Addresses

Address Offset	Read		Write	
0x0	**TDCH0** **STAT0** **SPI0**	Mod0 GP2 data Bits 16 to 31 Mod0 GP2 status Bits 0 to 15 Mod0 GP2 SPI control Bits	**TDCH0** **DAC0**	Mod0 GP2 data Bits 8 to 23 Vio,Vcc data
0x2	**TDCH1** **STAT1** **SPI1**	Mod1 GP2 data Bits 16 to 31 Mod1 GP2 status Bits 0 to 15 Mod1 GP2 SPI control Bits	**TDCH1** **DAC1**	Mod1 GP2 data Bits 8 to 23 Vio,Vcc data
0x4	**TDCL0**	Mod0 GP2 data Bits 0 to 15	**TDCL0**	Mod0 GP2 data Bits 0 to 7
0x6	**TDCL1**	Mod1 GP2 data Bits 0 to 15	**MC0**	Mod0 Control
0x8	**MBS**	Motherboard status register	**TDCL1**	Mod1 GP2 data Bits 0 to 7
0xA		n.a.	**MC1**	Mod1 Control
0xC		n.a.		n.a.

2.1.2 Register Structure

Write Registers

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
TDCHx	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8
TDCLx	D7	D6	D5	D4	D3	D2	D1	D0	-	-	-	-	-	-	-	-
DACx	Vio7	Vio6	Vio5	Vio4	Vio3	Vio2	Vio1	Vio0	Vc7	Vc6	Vc5	Vc4	Vc3	Vc2	Vc1	Vc0
MCx	AdrM	AdrM	AdrM	AdrM	AdrM	AdrM	AdrM	AdrM	Opc 7	Opc 6	Opc 5	Opc 4	Opc 3	Opc 2	Opc 1	Opc 0

Read Registers

Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
TDCHx	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
TDCLx	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Statx	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
SPIx	0	0	0	0	0	0	0	0	Reg1 (23)	Reg1 (22)	Reg1 (21)	Reg1 (20)	Reg1 (19)	Reg1 (18)	Reg1 (17)	Reg1 (16)
MBS	-	-	INTI1	WI1	RI1	-	-	-	-	-	INT0	WI0	RI0	-	-	-

2.1.3 Registers in Detail

MOD0 Registers

TDCH0	D23 to D8	Write: data to be written into TDC-GP2 registers,	High Word
TDCL0	D7 to D0		Low Word
TDCH0	D31 to D16	Read: read from TDC-GP2 registers,	High Word
TDCL0	D15 to D0		Low Word

DAC0	D15 to D8	Write access to DAC configuration register for setting Vio supply voltage
DAC0	D7 to D0	Write access to DAC configuration register for setting Vcc supply voltage Voltage V = 2 * 0.0196 * DAC[]. Example: 0x544C sets Vio = 3.3 V and Vcc = 3.0 V

MC0	D15 to D8	Control register to set the address mode AdrM	
		AdrM:	0x00 Setting to execute TDC-GP2 operation code.
			0x0F Enables access to D/A-Converter
			0xAA Power-on reset TDC-GP2
MC0	D7 to D0	TDC-GP2 operation code, executed by setting AdrM 0x00. For operation code details please refer to TDC-GP2 datasheet	

STAT0	D12	Error short, indicates a shorted sensor at temperature port
	D11	Error open, indicates an open sensor at temperature port
	D10	Indicates an overflow of the 14-Bit precounter in measurement range 2
	D9	Indicates a Timeout of the TDC
	D8 to D6	Number of hits on channel 2
	D5 to D3	Number of hits on channel 1
	D2 to D0	Pointer to the next free result register

SPI0	D7 to D0	TDC-GP2 REG_1 content., to be used for testing the communication.
SPI0	D15 to D8	Not relevant, default setting = “0”

MBS	D5 to D3	Motherboard Status register	
		WI0:	Indicates FPGA interrupt after finishing a write cycle, Low active
		RI0:	Indicates FPGA interrupt after finishing a read cycle, Low active
		INT0:	Indicates TDC interrupt, Low active

MOD1 Registers

TDCH1	D23 to D8	Write: data to be written into TDC-GP2 registers,	High Word
TDCL1	D7 to D0		Low Word
TDCH1	D31 to D16	Read: read from TDC-GP2 registers,	High Word
TDCL1	D15 to D0		Low Word

DAC1	D15 to D8	Write access to DAC configuration register for setting Vio supply voltage
DAC1	D7 to D0	Write access to DAC configuration register for setting Vcc supply voltage Voltage V = 2 * 0.0196 * DAC[]. Example: 0x544C sets Vio = 3.3 V and Vcc = 3.0 V

MC1	D15 to D8	Control register to set the address mode AdrM	
		AdrM:	0x00 Setting to execute TDC-GP2 operation code.
			0x0F Enables access to D/A-Converter
			0xAA Power-on reset TDC-GP2
MC1	D7 to D0	TDC-GP2 operation code, executed by setting AdrM 0x00. For operation code details please refer to TDC-GP2 datasheet	

STAT1	D12	Error short, indicates a shorted sensor at temperature port.
	D11	Error open, indicates an open sensor at temperature port
	D10	Indicates an overflow of the 14-Bit precounter in measurement range 2
	D9	Indicates a Timeout of the TDC
	D8 to D6	Number of hits registered on channel 2
	D5 to D3	Number of hits registered on channel 1
	D2 to D0	Pointer to the next free result register

SPI1	D7 to D0	TDC-GP2 REG_1 content, to be used for testing the communication.
SPI1	D15 to D8	Not relevant, default setting = "0"
MBS	D13 to D11	Motherboard Status register
		WI1 Indicates FPGA interrupt after finishing a write cycle, Low active
		RI1 Indicates FPGA interrupt after finishing a read cycle, Low active
		INT1 Indicates MOD1 FPGA interrupt, Low active

2.2 Read-/Write-access

Read-/write-access to the ATMD is controlled by an FPGA. The PC communicates with the FPGA using a 16-Bit parallel bus. The FPGA communication with the TDC-GP2 is by means of the serial SPI interface. As the TDC-GP2 registers are 24 bit wide, each read-/write-cycle from or to the TDC is split in read-/write commands.

2.2.1 Write

The format of a write command depends on the address mode:

AdrM = 0xAA

One word command

Example: Power-on reset

```
_outpw(base+MC,0xAA00);Sleep(1);
```

AdrM = 0x0F

Two word command, completed by a write to address DACx

Example: Writes to DAC, setting Vio = Vcc = 3.0V

```
_outpw(base+MC,0x0F00);_outpw(base+TDCHw,0x4c4c);Sleep(10);
```

AdrM = 0x00

Three word command, completed by 8 Bit opcode and two writes to address TDCHx and TDCLx

Example: Write 0x113432 to TDC-GP2 register 0

```
_outpw(base+MC,0x0080);_outpw(base+TDCHw,0x1134);
_outpw(base+TDCLw,0x3200);while((_inpw(base+MBS)&0x10)==0x10);
// The while loop checks when the writing is finished
```

2.2.2 Read

The format of a write command depends on data to be read:

MBSx One read command to get the motherboard status register content

Example: Check interrupt flag

```
while((_inpw(base+MBS)&0x20)==0x20);
```

SPIx One write command (opcode 00B5), one read command to get the SPI = TDC-GP2 REG_1 content

Example:

```
outpw(base+MC,0x00B5);test=_inpw(base+SPIO);
```

STATx One write command (opcode 00B4), one read command to get the TDC-GP2 status

Example:

```
outpw(base+MC,0x00B4);timeout=(_inpw(base+STAT0)&0x0600);
```

TDCH/Lx One write command and two read commands to get the TDC-GP2 result

Example: Read RES_0:

```
_outpw(base+MC, 0x00B0);              // Opcode read address 0
while((_inpw(base+MBS)&0x8)==0x8);  // Wait for FPGA end of read TDC
result=_inpw(base+TDCH)<<16;
result=_inpw(base+TDCL)+result;
```

The following example shows how to write software for the ATMD-GP2.

2.3 Measurement Range 1

The ATMD-GP2 is configured for a simple time interval measurement between START and STOP1 in measurement range 1 without autocalibration.

1. Get base address of the ATMD-PCI interface card

```
#include "atmd_pci.h"

// detect ATMD PCI (call GetATMDPCIBoardCount()
// to detect number of ATMD-PCI boards)

iBoardCount = GetATMDPCIBoardCount();
printf("No. of ATMD PCI boards found = %d\n",iBoardCount);

i=0;
while((!AtmdOK) || (i>4))
{
      AtmdOK = GetATMDPCIBaseAddr(i,dwTemp);
      atmd_pci_base_address[i] = (WORD) dwTemp;
      i++;
}
if (AtmdOK)
{
      printf("ATMD PCI Board found on 0x%x\n",atmd_pci_base_address[i-1]);
      base = atmd_pci_base_address[i-1];              // base = base address
      if(!EnablePortAccess())
      {
            AfxMessageBox("Giveio.sys couldn't be opened");
      }
}
else
{
      AfxMessageBox("ATMD-PCI interface not found");
}
```

2. Address settings for slot 0 or slot 1

```
//  ******************** Address settings **************
Slot = 0;               //manual setting of slot, 0 or 1

if (Slot==0)            //MOD0
{
      TDCHw=0; TDCLw=4; TDCHr=0; TDCLr=4; DAC=0; MC=6; STAT=0; SPI=0; MBS=8;
}
else                   //MOD1
{
      TDCHw=2; TDCLw=8; TDCHr=2; TDCLr=6; DAC=2; MC=10; STAT=2; SPI=2;
      MBS=8;
}
```

2. Board Reset

```
//  ******************** board-reset ******************

_outpw(base+MC,0xAA00);Sleep(1);
```

3. Configuration of Vcc and Vio supply voltages by setting the DACx register

```
//  ******************** Set Voltage ******************
fVcc = 3.0;
```

```
fVio = 3.0;

Vcc = WORD(fVcc / 2 / 0.0196); if (Vcc>0x5B) Vcc=0x5B;
Vio = WORD(fVio / 2 / 0.0196)<<8; if (Vio>0x7F00) Vio=0x7F00;

_outpw(base+MC,0x0F00);_outpw(base+TDCHw,Vio+Vcc);Sleep(10); //Writes to DAC
```

4. Set the TDC-GP2 control registers

```
// ****************** Setup ***************************
//
//            OpCode + address        Data 23 to 8             Data 7 to 0
//                  ||                    ||||                     ||
_outpw(base+MC,0x0080);_outpw(base+TDCHw,0x0034);_outpw(base+TDCLw,0x3000);
while((_inpw(base+MBS)&0x10)==0x10); // write cycle finished

// Reg 0: Enable rising edge, measurement range 1, ClkHSDiv = 4,
// autocalibration off, start oscillator

_outpw(base+MC,0x0081);_outpw(base+TDCHw,0x0101);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 1: 1. Stop channel 1 - Start, # of hits on channel 1 = 1, # of hits on
// channel 2 = 0

outpw(base+MC,0x0082);_outpw(base+TDCHw,0x2000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 2: ALU interrupt enable

_outpw(base+MC,0x0083);_outpw(base+TDCHw,0x1800);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 3: DELVAL 2 = 0, Fast interrupt disable, Predivider for Timeout = 0

_outpw(base+MC,0x0084);_outpw(base+TDCHw,0x2000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 4: Default settings

_outpw(base+MC,0x0085);_outpw(base+TDCHw,0x0000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 5 Default settings
```

The ATMD-GP2 is configured to operate in to measurement range 1 with sensitivity to rising edge. The ALU calculates the time interval between the first hit on STOP 1-channel and the start event on START-input.

5. Measurement

```
//******************** Measurement ********************

_outpw(base+MC, 0x0004);Sleep(10);              // Separate calibration TDC
do
{
      _outpw(base+MC, 0x0070);                  // Initialize TDC
      while((_inpw(base+MBS)&0x20)==0x20);      // Wait for interrupt TDC

      _outpw(base+MC, 0x00B4);                  // Read gp2 status register
      while((_inpw(base+MBS)&0x8)==0x8);        // Wait for end of read TDC

      _outpw(base+MC, 0x00B0);                  // OPCode read address 0
```

```
      while((_inpw(base+MBS)&0x8)==0x8);        // Wait for end of read TDC

      result=_inpw(base+TDCHr)<<16;
      result=_inpw(base+TDCLr)+result;
      printf("%X %5.3f ns\n",result, float(result)/65536*250*4);

// Keyboard: "i" = interrupt measurement, "q" = quit measurement
      if(kbhit())
      {
            ch=getch();
            if(ch=='q')
                  quit=true;

            if(ch=='i')
            {
                  while(!kbhit());
                  quit=false;
            }
      }
} while ( !quit );
```

The ATMD is configured to measure without autocalibration. Therefore a separate calibration run of the TDC is initialized by sending opcode 0x04 before the measurement routine starts. After initializing the TDC the program loop checks the INT0 flag of the MBS-register for the end of a measurement cycle. Then the measurement value is read by executing a read cycle to ATMD read registers TDCH0 and TDCL0. The measurement routine runs in an endless loop. Pushing “i”-key will interrupt the measurement, Pushing the “i”-key several times executes a single measurement for each operation. A “q” quits the measurement program.

2.4 Measurement Range 2

The ATMD-GP2 is configured for a simple time interval measurement between START and STOP1 in measurement range 2 with autocalibration.

1. to 3. Please refer to previous section

4. Set the TDC-GP2 control registers

```
// ****************** Setup ****************************
//
//            OPCode + address        Data 23 to 8              Data 7 to 0
//                  ||                      ||||                    ||
_outpw(base+MC,0x0080);_outpw(base+TDCHw,0x0004);_outpw(base+TDCLw,0x2800);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 0: Enable rising edge, measurement range 2, autocalibration on, start
// oscillator

_outpw(base+MC,0x0081);_outpw(base+TDCHw,0x2142);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 1: 1. Stop channel 1 – Start # of hits on channel 1 = 2,
// # of hits on channel 2 = 0

_outpw(base+MC,0x0082);_outpw(base+TDCHw,0xE000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 2: all interrupt sources enabled

_outpw(base+MC,0x0083);_outpw(base+TDCHw,0x1800);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 3: Default Settings, Predivider for Timeout MR2 = 1 (256µs)
```

```
_outpw(base+MC,0x0084);_outpw(base+TDCHw,0x2000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);
// Reg 4: Default settings

_outpw(base+MC,0x0085);_outpw(base+TDCHw,0x0000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 5 Default settings
```

The ATMD is set to measurement range 2 with sensitivity to rising edges. The autocalibration function is enabled and the ALU calculates the time interval between the first stop on channel 1 and the start event. Please note that the ALU settings HITT1 and HIT2 are modified in measurement range 2.

5. Measurement

```
//******************** Measurement ********************
do
{
      _outpw(base+MC, 0x0070);                  // Initialize TDC
      while((_inpw(base+MBS)&0x20)==0x20);      // Wait for interrupt TDC

      _outpw(base+MC, 0x00B4);                  // Read gp2 status register
      while((_inpw(base+MBS)&0x8)==0x8);        // Wait for end of read TDC

      _outpw(base+MC, 0x00B0);                  // OPCode read address 0
            while((_inpw(base+MBS)&0x8)==0x8);  // Wait for end of read TDC

      result=_inpw(base+TDCHr)<<16;
      result=_inpw(base+TDCLr)+result;
      printf("%X %5.3f\n",result, float(result)/65536*250);

// Keyboard: "i" = interrupt measurement, "q" = quit measurement
      if(kbhit())
            {
                  ch=getch();
                  if(ch=='q')
                        quit=true;

                  if(ch=='i')
                  {
                        while(!kbhit());
                        quit=false;
                  }
            }

} while ( !quit );
```

The ATMD-GP2 is configured to measure with autocalibration mode. Therefore a separate calibration run of the TDC is not necessary. After initializing the TDC the program loop checks the INT0 flag of the MBS-register for the end of a measurement cycle. Then the measurement value is read by executing a read cycle to ATMD read register TDCH0 and TDCL0. The measurement routine runs in an endless loop. Pushing “i”-key will interrupt the measurement, Pushing the “i”-key several times executes a single measurement for each operation. A “q” quits the measurement program.

2.5 Special Functions

This section provides sample programs for the use of the special functions of the ATMD-GP2.

2.5.1 Temperature Measurement

The ATMD-GP2 provides a temperature measuring unit with 4 ports. The sensors require a minimum resistance of 500 Ohm. The following sample program shows the configuration of the TDC-GP2 temperature unit. The configuration steps 1. to 3. will be the same as described in section 2.2.

4. Set the ATMD-MGP2 control registers

```
// ****************** Setup ***************************
//
//           OPCode + address          Data 23 to 8              Data 7 to 0
//                  ||                      ||||                      ||
_outpw(base+MC, 0x0080);_outpw(base+TDCHw,0x0007);_outpw(base+TDCLw,0x6000);
while((_inpw(base+MBS)&0x10)==0x10);

// Start CLKHS, 4 temperature ports, Tcycle = 300µs @ 4MHz ref. clock, 2 fake
// measurements, SelClkT = 1,

_outpw(base+MC,0x0081);_outpw(base+TDCHw,0x0040);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// No Hits

_outpw(base+MC,0x0082);_outpw(base+TDCHw,0xE000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Default Values, all interrupt sources enabled

_outpw(base+MC,0x0083);_outpw(base+TDCHw,0x1800);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

 // Default settings
      _outpw(base+MC,
0x0084);_outpw(base+TDCHw,0x2000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Default
_outpw(base+MC,0x0085);_outpw(base+TDCHw,0x0000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Default settings
```

The above program section configures the ATMD-GP2 to operate with 4 temperature ports. The cycle time for the temperature measurement is 300 µs, as recommended in the TDC-GP2 datasheet. Two FAKE-measurements are introduced at the beginning of each measurement cycle.

5. Measurement

```
//******************** Measurement ********************
do
{
      _outpw(base+MC, 0x0002);              // Start Temp. measurement
      while((_inpw(base+MBS)&0x20)==0x20); // wait for interrupt Temp.-
                                            // measurement finished

      _outpw(base+MC, 0x00B4);              //Read gp2 status register
      while((_inpw(base+MBS)&0x8)==0x8);    // Wait for end of read TDC

      // Read temperature ports
      // port 0
```

```
        _outpw(base+MC, 0x00B0);                // OPCode read address 0
        while((_inpw(base+MBS)&0x8)==0x8);      // Wait for end of read TDC

        result=_inpw(base+TDCHr)<<16;
        result=_inpw(base+TDCLr)+result;
        printf("Port 0:%X %5.3f\n",result, float(result)/65536*250*4);

        // port 1
        _outpw(base+MC, 0x00B1);                // OPCode read address 1
        while((_inpw(base+MBS)&0x8)==0x8);      // Wait for end of read TDC

        result=_inpw(base+TDCHr)<<16;
        result=_inpw(base+TDCLr)+result;
        printf("Port 1:%X %5.3f\n",result, float(result)/65536*250*4);

        // port 2
        _outpw(base+MC, 0x00B2);                // OPCode read address 2
        while((_inpw(base+MBS)&0x8)==0x8);      // Wait for end of read TDC

        result=_inpw(base+TDCHr)<<16;
        result=_inpw(base+TDCLr)+result;
        printf("Port 2:%X %5.3f\n",result, float(result)/65536*250*4);

        // port 3
        _outpw(base+MC, 0x00B3);                // OPCode read address 3
        while((_inpw(base+MBS)&0x8)==0x8);      // Wait for end of read TDC

        result=_inpw(base+TDCHr)<<16;
        result=_inpw(base+TDCLr)+result;
        printf("Port 3:%X %5.3f\n",result, float(result)/65536*250*4);

        // Keyboard: "i" = interrupt measurement, "q" = quit measurement
        if(kbhit())
        {
              ch=getch();
                    if(ch=='q')
                          quit=true;

                    if(ch=='i')
                    {
                          while(!kbhit());
                          quit=false;
                    }
        }
} while ( !quit );
```

The temperature measurement is started by sending opcode 0x0002. All 4 temperature ports are scanned fully automated. After finishing the measurement the interrupt flag is set and measurement values are read from result registers TDCH0 and TDCL0. Pushing “i”-key will interrupt the measurement, Pushing it several times executes single temperature measurement cycles for all temperature ports. A “q” quits the measurement program.

2.5.2 Fire Pulse Generator

This sample program shows the use of the fire pulse generator generating a defined pulse sequence. The configuration steps 1. to 3. are the same as described in section 2.2.

4. Set the ATMD-GP2 control registers

```
// ****************** Setup ***************************
//
//          OPCode + address          Data 23 to 8              Data 7 to 0
//                 ||                      ||||                      ||
_outpw(base+MC,0x0080);_outpw(base+TDCHw,0x1104);_outpw(base+TDCLw,0x2800);
while((_inpw(base+MBS)&0x10)==0x10);

// Fire# = 1, Div_Fire = 1, autocalibration; MR2. clock calibration and temp.-
// measurement not configured

_outpw(base+MC,0x0081);_outpw(base+TDCHw,0x2142);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 1: 1. Stop channel 1 - Start, # of hits on channel 1 = 2

_outpw(base+MC,0x0082);_outpw(base+TDCHw,0xA000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 2: all interrupts enable, edge sensitivity falling and rising edge

_outpw(base+MC,0x0083);_outpw(base+TDCHw,0x1000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 3: DELVAL 2 = 0, Fast interrupt disable, Predivider for Timeout = 0

_outpw(base+MC,0x0084);_outpw(base+TDCHw,0x2000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 4: DELVAL 3 = 0

_outpw(base+MC,0x0085);_outpw(base+TDCHw,0x0000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Reg 5: Send one Fire pulse with no pulse repetition
```

The ATMD-GP2 runs in measurement range 2, measuring the time interval between the first hit on Stop 1 and the Start. The fire generator is configured to send a single pulse without pulse repetition.

5. Measurement

```
//******************** Measurement ********************
do
{
    _outpw(base+MC, 0x0070);              // Initialize TDC
    while((_inpw(base+MBS)&0x10)==0x10);  // wait for end write TDC

    _outpw(base+MC, 0x0001);              // Start_Cycle (Fire)
    while((_inpw(base+MBS)&0x10)==0x10);  // wait for end write TDC

    while((_inpw(base+MBS)&0x20)==0x20);  // wait for interrupt

    _outpw(base+MC, 0x00B4);              // Read gp2 status register
    while((_inpw(base+MBS)&0x8)==0x8);    // wait for end of read TDC

    _outpw(base+MC, 0x00B0);              // OPCode read address 0
    while((_inpw(base+MBS)&0x8)==0x8);    // wait for end of read TDC
```

```
    result=_inpw(base+TDCHr)<<16;
    result=_inpw(base+TDCLr)+result;
    printf("%X %5.3f\n",result, float(result)/65536*250);

    // Keyboard: "i" = interrupt measurement, "q" = quit measurement
    if(kbhit())
    {
            ch=getch();
            if(ch=='q')
                  quit=true;

            if(ch=='i')
            {
                  while(!kbhit());
                  quit=false;
            }
      }
} while ( !quit );
```

After initializing the opcode 0x01 triggers the fire-pulse generator. The TDC unit waits for Start and Stop events and finally sets the interrupt flag when the data are available.
Pushing “i”-key will interrupt the measurement, A “q” quits the measurement program.

2.5.3 Clock Calibration Unit

The clock calibration unit allows the use of ceramic resonators with their poor tolerances. Therefore the periods of the high speed oscillator are measured for a specific time interval and compared to a theoretical value. The ratio between measured number of clock periods and the theoretical number can be used in the microcontroller to correct for a wrong ceramic resonator's frequency. The specific time interval is based on a defined number of periods of the 32.768 kHz oscillating quartz signal and is configured by CALRES#.

Step 1. to 3. Please refer to section 2.2.

4. Set the TDC-GP2 control registers

```
// ****************** Setup ****************************
//
//           __OPCode + address      ____Data 23 to 8          __Data 7 to 0
//                 ||                      ||||                      ||
_outpw(base+MC,0x0080);_outpw(base+TDCHw,0x00C4);_outpw(base+TDCLw,0x2800);
while((_inpw(base+MBS)&0x10)==0x10);

// MR2,autocalibration, start oscillator, ClkHSDiv = 0, CalRes# = 3

_outpw(base+MC,0x0081);_outpw(base+TDCHw,0x2142);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Stop 1 Hit#1 - Start,0 Hits on stop channel 2, 2 Hits on stop channel 2

_outpw(base+MC,0x0082);_outpw(base+TDCHw,0xE000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// All interrupt sources enabled

_outpw(base+MC,0x0083);_outpw(base+TDCHw,0x1000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);

// Default settings

_outpw(base+MC,0x0084);_outpw(base+TDCHw,0x2000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);
// Default settings
```

```
_outpw(base+MC,0x0085);_outpw(base+TDCHw,0x0000);_outpw(base+TDCLw,0x0000);
while((_inpw(base+MBS)&0x10)==0x10);
// Default settings
```

The ATMD-GP2 operates in measurement range 2, measuring the time interval between the first hit on Stop 1 channel and the Start event. The clock calibration unit is configured to use 16 periods of the 32,768khz clock signal. (CALRES# = 4), that represents a reference value of 488,28125 µs for the time interval.

5. Measurement

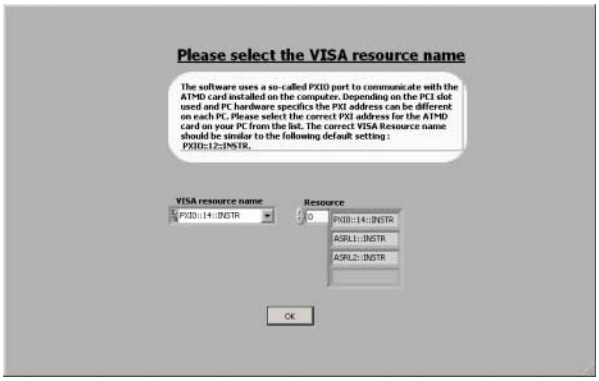
```
//******************** Measurement ********************
do
{
      _outpw(base+MC, 0x0070);            // Initialize TDC
      while((_inpw(base+MBS)&0x20)==0x20);// Wait for interrupt TDC

      //Clock calibration
      _outpw(base+MC, 0x0003);            // Start Calibration Resonator
      while((_inpw(base+MBS)&0x20)==0x20);// Wait for interrupt TDC

      _outpw(base+MC, 0x00B0);            // Read Clock Calibration Value
      while((_inpw(base+MBS)&0x8)==0x8);  // Wait for end of read TDC

      result=_inpw(base+TDCHr)<<16;
      result=_inpw(base+TDCLr)+result;
      fresult=float(result)/65536.0;      // measured Value for time interval

      corr=488281.25/fresult/250.0;       // calculate correction factor

      // Time measurement
      _outpw(base+MC, 0x0070);            // Initialize TDC
      while((_inpw(base+MBS)&0x20)==0x20);// Wait for interrupt TDC

      _outpw(base+MC, 0x00B4);            // Read gp2 status register
      while((_inpw(base+MBS)&0x8)==0x8);  // Wait for end of read TDC
      printf("GP2_stst: %X\n",_inpw(base+MBS));

      _outpw(base+MC, 0x00B0);                  // OPCode read address 0
      while((_inpw(base+MBS)&0x8)==0x8);        // Wait for end of read TDC

      result=_inpw(base+TDCHr)<<16;
      result=_inpw(base+TDCLr)+result;
      printf("%X %5.3f\n",result, float(result)/65536*250*corr);

      // Keyboard: "i" = interrupt measurement, "q" = quit measurement
      if(kbhit())
      {
            ch=getch();
            if(ch=='q')
                  quit=true;

            if(ch=='i')
            {
                  while(!kbhit());
                  quit=false;
            }
      }
} while ( !quit );
```

After a first initialization of the TDC-GP2 the program starts the clock calibration process. Based on the reference time interval of 488, 28125 µs the number of periods of the 4 MHz clock within this time interval is measured. The measured value is compared to the theoretical value of 488.28125 µs/250 ns.

3 ATMD_GP2 Measurement Software

3.1 Measurement Software

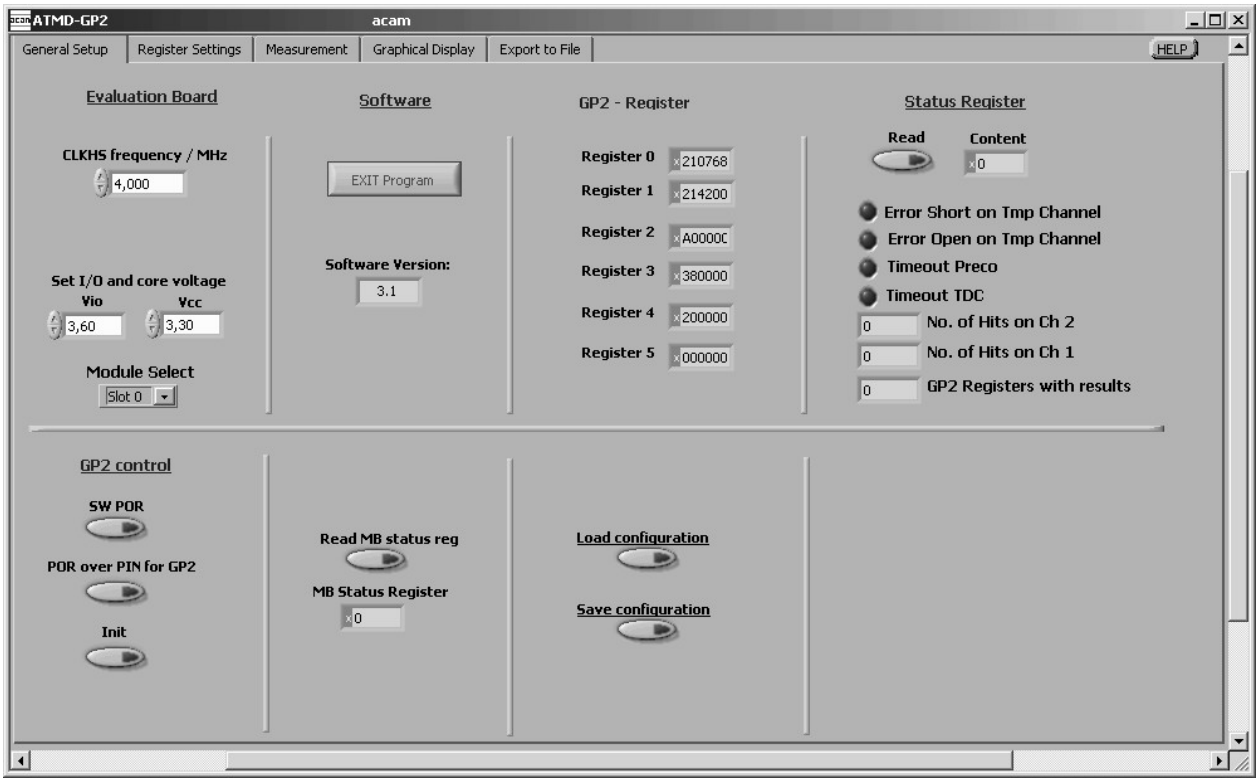
When starting the ATMD software the user is first asked to select the right PCI interface card for communication:



Please select a PXI... device and press “OK”.

After that the first page of the ATMD-GP2 measurement and configuration software appears. This page shows the general setup items.

3.2 General Setup



Evaluation Board

First select the slot of the AM-GP2 module. The ATMD-Motherboard provides two slots, “Slot 0” at the right and “Slot 1” at the left. In principle the ATMD-GP2 can operate with two AM-GP2 modules, referred to as “MOD0” and “MOD1”. Please note that one instance of the ATMD-software does support only one AM-GP2 module.

"CLKHS frequency" indicates the frequency of the high speed oscillator which is 4 MHz. In case this oscillator is replaced by another frequency type the value can be adjusted in this field. Any result displays in time units refer to this value.

Further the ATMD-GP2 allows to control the core supply voltage Vcc and the I/O-supply voltage Vio by software.

Software
Shows the version of the configuration software and provides a button to exit the program.

GP2 Register content
Displays the content of the TDC-GP2 write registers. For a detailed description please refer to TDC-GP2 datasheet.

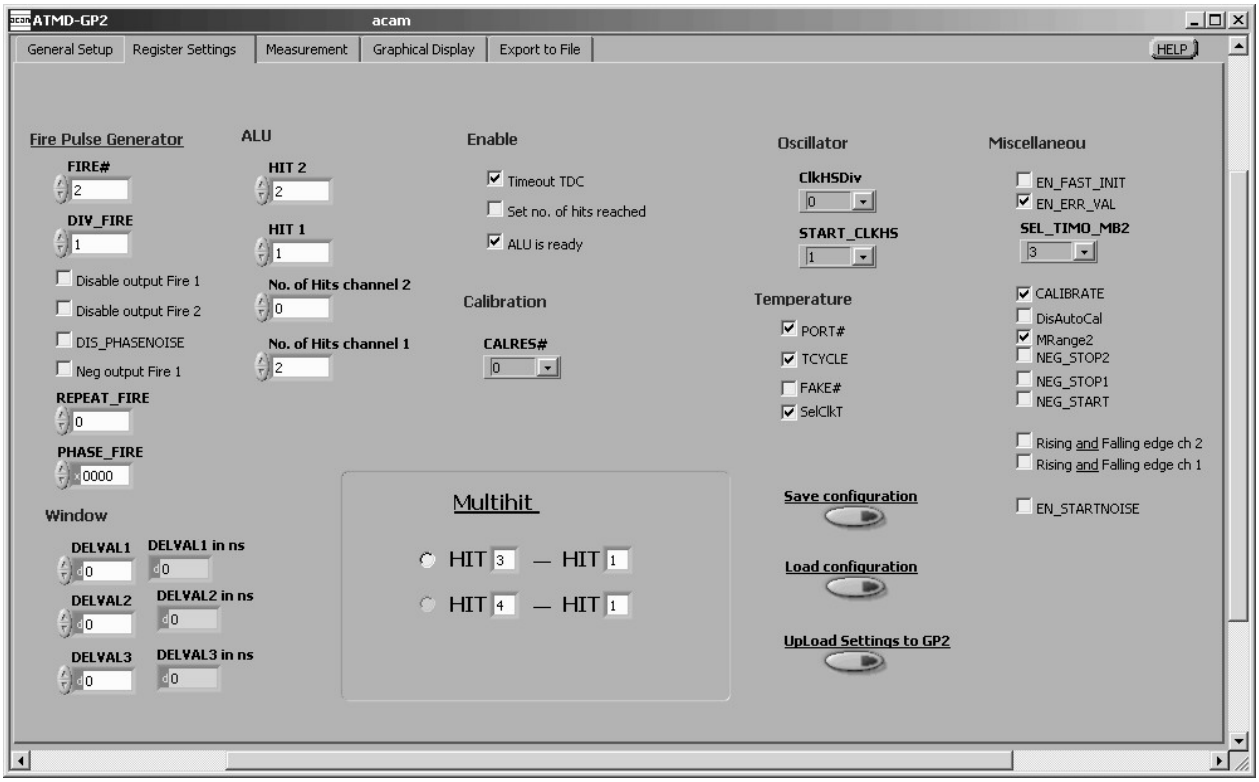
Status register
This section provides information about the TDC-GP2 status register and indicates whether an overflow or an error in the temperature measurement occurred. For further information please have a look at the TDC-GP2 datasheet, chapter 2.6.3.

GP2 Control
Pressing the button "SW POR" executes a software power-on reset by sending opcode 0xA0. "POR over PIN 2" generates a hardware based power-on reset on Pin 2 of the TDC-GP2. "Init" activates the TDC and prepares it for the next measurement.

Read MB status register
Displays the content of the ATMD-GP2 Motherboard status register MBS. See also chapter 2.1.3.

Load/Save configuration
It is possible to save current settings or to upload older configurations. The file extension is .cfg.

3.3 Register Settings



This page is for the application specific configuration of the TDC-GP2. Also here the configuration can be saved or older configurations can be loaded. Pressing the button "Upload settings to GP2" will overwrite the configuration of the TDC-GP2 to current settings.

Fire Pulse Generator

This section is used to configure the fire pulse generator. “Fire#” sets the number (0 to 15) of pulses for each sequence. “Div_Fire” is a divider (1 to 15) for the reference clock. The fire pulse frequency is calculated as high speed clock *2 / (Div_Fire+1). With “Repeat_Fire” the repetition rate of the pulse sequences is set. ”PHFIRE” represents a HEX-coded 16-Bit value that defines the phase of each pulse within a pulse sequence. Each phase of the up to 15 pulses is defined by setting “0” or “1” to the single bits.
Dis_PhaseNoise = "1" disables a phase noise unit for the fire pulse generator that is needed to provide the statistics for correct averaging.

ALU Configuration

Defines the number of expected hits (HITINx) and the calculation rule (Hitx) for the ALU.
In measurement range 1 the calculation algorithm is HIT1 – HIT2.

HIT1:	HIT2:
0 = Start	0 = Start
1 = 1. Stop Ch1	1 = 1. Stop Ch1
2 = 2. Stop Ch1	2 = 2. Stop Ch1
3 = 3. Stop Ch1	3 = 3. Stop Ch1
4 = 4. Stop Ch1	4 = 4. Stop Ch1
5 = no action	5 = no action
6 = Cal1 Ch1	6 = Cal1 Ch1
7 = Cal2 Ch1	7 = Cal2 Ch1
9 = 1. Stop Ch2	9 = 1. Stop Ch2
A = 2. Stop Ch2	A = 2. Stop Ch2
B = 3. Stop Ch2	B = 3. Stop Ch2
C = 4. Stop Ch2	C = 4. Stop Ch2

In measurement range 2 will change this algorithm to: HIT2 – HIT1 where HIT1 = START.

HIT1:	HIT2:
1 = Start	2 = 1. Stop Ch1
	3 = 2. Stop Ch1
	4 = 3. Stop Ch1

Note:
When using measurement range 2 the start event has to be regarded as a hit on channel 1. So when you expect e. g. 1 stop hit on channel 1 you have to enable two hits!

Multihit Settings

The TDC-GP2 itself can store up to 4 hits for each channel in measurement range 1 and 3 hits for STOP1 channel in measurement range 2. But the ALU can do only one operation at once whereas the first calculation instruction is configured by the above settings. Enabling the multihit feature forces the ALU to execute up to 2 additional calculations, to be specified here.

Enable Interrupts

The user can select 3 independent sources for an interrupt, linked by an or function:
a. Timeout of the TDC
b. Set number of hits is reached
c. ALU is ready

Calibration Resonator

"CalRes#" sets the number of periods of the high speed clock, that are used for calibrating the ceramic oscillator.

Oscillator Control

“ClkHSDiv” sets the predivider for high speed clock. “Start_CLKS” switches on the ceramic oscillator and sets the start behavior of the ceramic oscillator.
Note:
Attend the bug report in section 7 of the TDC-GP2 datasheet when using Start_CLKHS!

Temperature Measurement

"Port#" configures the number of Ports used for temperature measurement. Activating this item enables the use of 4 ports, disabling provides only 2 temperature ports.

If "Tcycle" is selected the cycle time for a temperature measurement is set to 300 µs. Otherwise it is 150 µs (based on the 4 MHz clock as reference).

Activating "Fake#" sets the number of Fake measurements to 7. Otherwise 2 Fake measurements are executed before starting a temperature measurement.

With "SelClkT"= "0" the 32,768 kHz clock is used as reference for temperature measurement. With "SelClkT"= "1" the period for the cycle clock is 128 * CLKHS.

Miscellaneous

These settings are very important for the behavior of the ATMD-GP2.

"EN_FAST_INIT" enables the fast initialization mode of the TDC-GP2. The interrupt automatically initializes the TDC for the next measurement.

Setting "EN_ERR_VAL" forces the ALU to write "0xFFFFFFFF" to the output register in case of a timeout.

With "SEL_TIMO_MB2" the user defines a timer that generates a timeout after a specified time interval. This is most helpful in measurement range 2.

0 = 64 µs
1 = 256 µs
2 = 1.024 ms
3 = 4.096 ms

With "CALIBRATE" on the ALU calibrates the measurement results. With "CALIBRATE" off the ALU just transfers the raw data to the output. In measurement range 2 it is mandatory to set "CALIBRATE" on.

Activating "DisAutoCal" disables autocalibration after with each measurement.

MRange2 switches on measurement range 2.

The input sensitivity of each channel can be inverted by activating NEG_STOP1, NEG_STOP2 and / or NEG_START. With "Rising and Falling edge" the stop channels can be configured to be sensitive to both, rising and falling edge. for the referring channel.

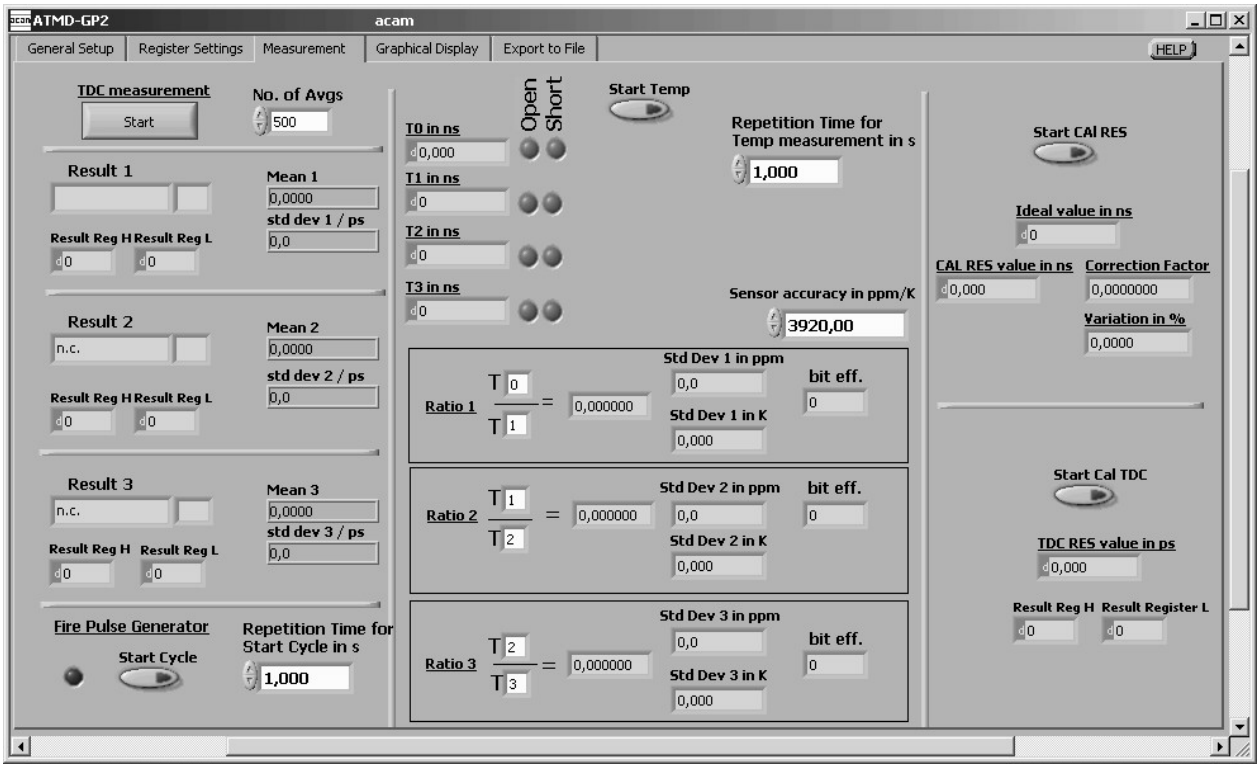
To improve statistics, especially when averaging is used, "EN_STARTNOISE" switches on a noise unit, that adds a random noise to the start channel. This is useful to improve the statistics for successful averaging when working with the fire pulse generator.

Window control

In measurement range 2 the TDC-GP2 offers the possibility to set 3 windows, one for each stop, for a precise stop enable. The TDC doesn't accept a hit in a time interval DEL_VAL1 after the start. A second hit is not accepted before DEL_VAL2, a third hit not before DEL_VAL3.

3.5 Measurement

The measurement sheet displays the measurement results of the ATMD-GP2 evaluation system.



TDC Measurement

Before starting the measurement the user has to set the requested averaging rate in "No. of Avgs".
Up to 3 multihit results are displayed, depending on the ALU configuration. “Mean x” displays the averaged values calculated over the number of measurements set in field “No. of Avgs”. “std dev...” shows the standard deviation of each single measurement in picoseconds.

Fire Pulse Generator

The Button “Start Cycle” starts the fire pulse generator by sending opcode “Start_Cycle” to the ATMD. The repetition time for sending the pulse sequences is adjustable and can be changed by pushing arrow up / down button or inserting the requested value in the description field. The emission of each pulse sequence by the pulse fire generator is indicated by a short blinking of the green control lamp.

Temperature Measurement

The values in the middle section of the “Measurement”-sheet all refer to the temperature measurement unit. After setting the repetition time and the value for sensor accuracy the temperature measurement is started by a mouse click on the “Start Temp”-Button. T0 to T4 display time values in nanoseconds that directly refer to the temperature value measured on the selected temperature ports.

The software also calculates the ratios of two temperature ports, the appropriate standard deviations in ppm and Kelvin and the number of effective bits. The software is capable of calculating up to three different ratios.

Resonator Calibration

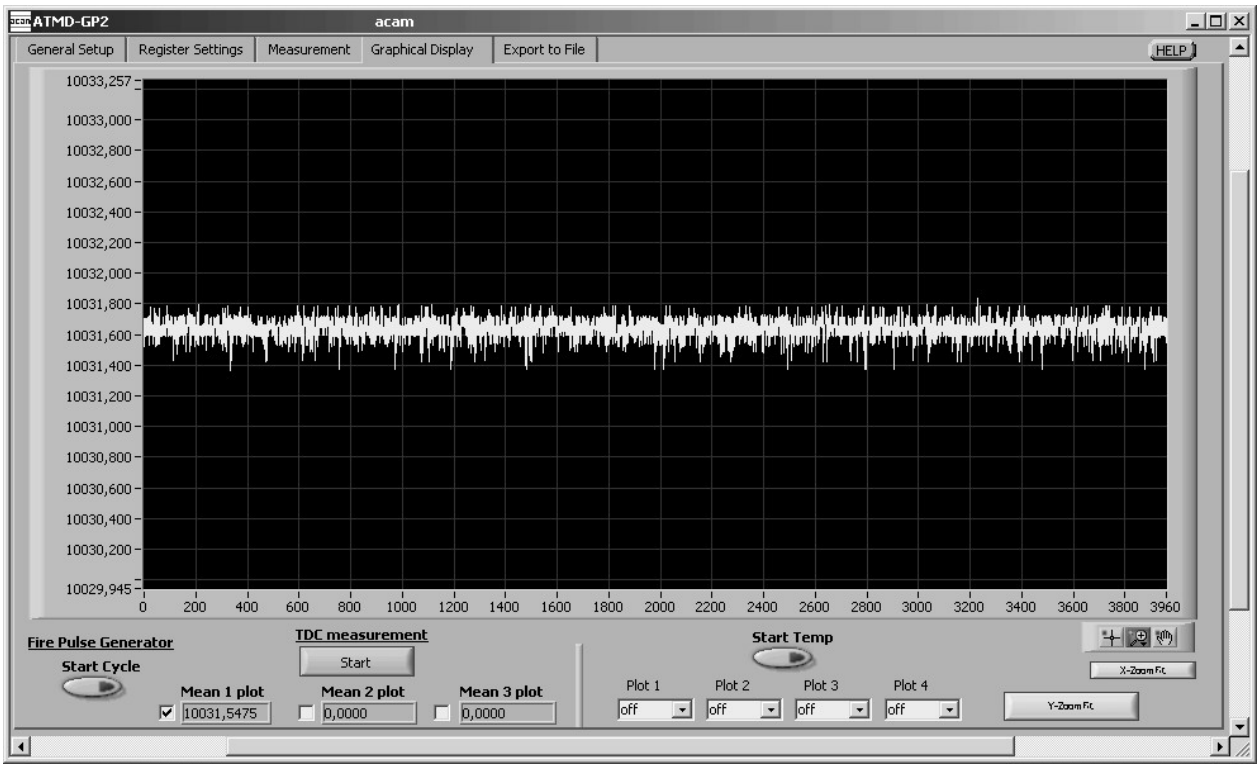
“Start CAL RES” runs the calibration of the ceramic high-speed oscillator by sending the appropriate opcode to the ATMD. The calculated correction factor is displayed. This value can be used to correct for a deviation in oscillator frequency from the nominal value. Detailed information about the clock calibration can be found in the TDC-GP2 datasheet, chapter 5.1.3 and 5.1.4.

TDC Calibration

“Start CAL TDC” forces a calibration run of the TDC-GP2 and shows the resolution of the Time-to-Digital Converter in picoseconds. Further the raw data content of the TDC-GP2 result register is shown.

3.6 Graphical Display

This page is for the graphical display, showing the measurement results (y) over runtime (x).
The scales can be modified directly by editing the corner values or by using the magnifying glass tool.

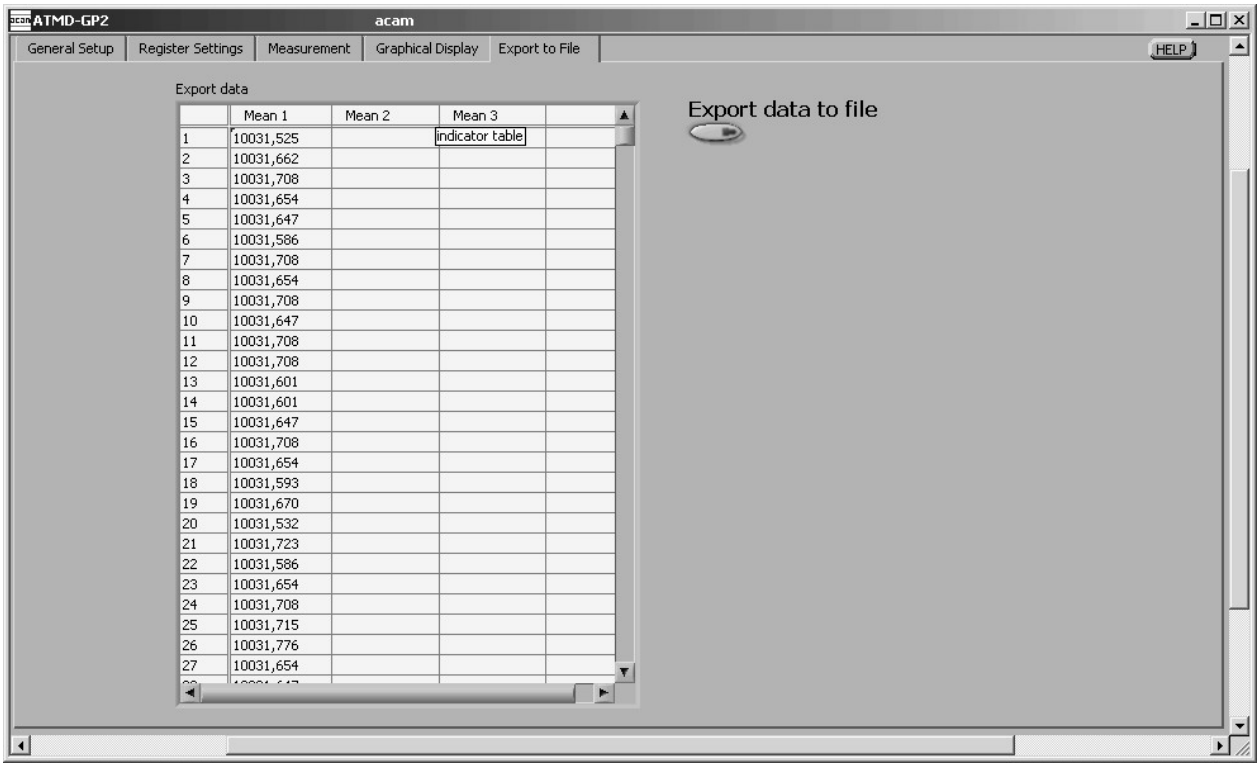


"Start Cycle" starts the fire pulse generator, Pushing the "Start"-Button runs the time measurement and shows the measurement values. The temperature measurement is started by means of the "Start Temp"-button. Up two four temperature channels can be displayed (Plot 1 to Plot 4) whereas the user can select to display the temperature value or the ratio of two temperature ports.

Note: The software handles the temperature measurement with higher priority than the time interval measurement. In case both are active only the temperature results will be displayed.

3.7 Export to File

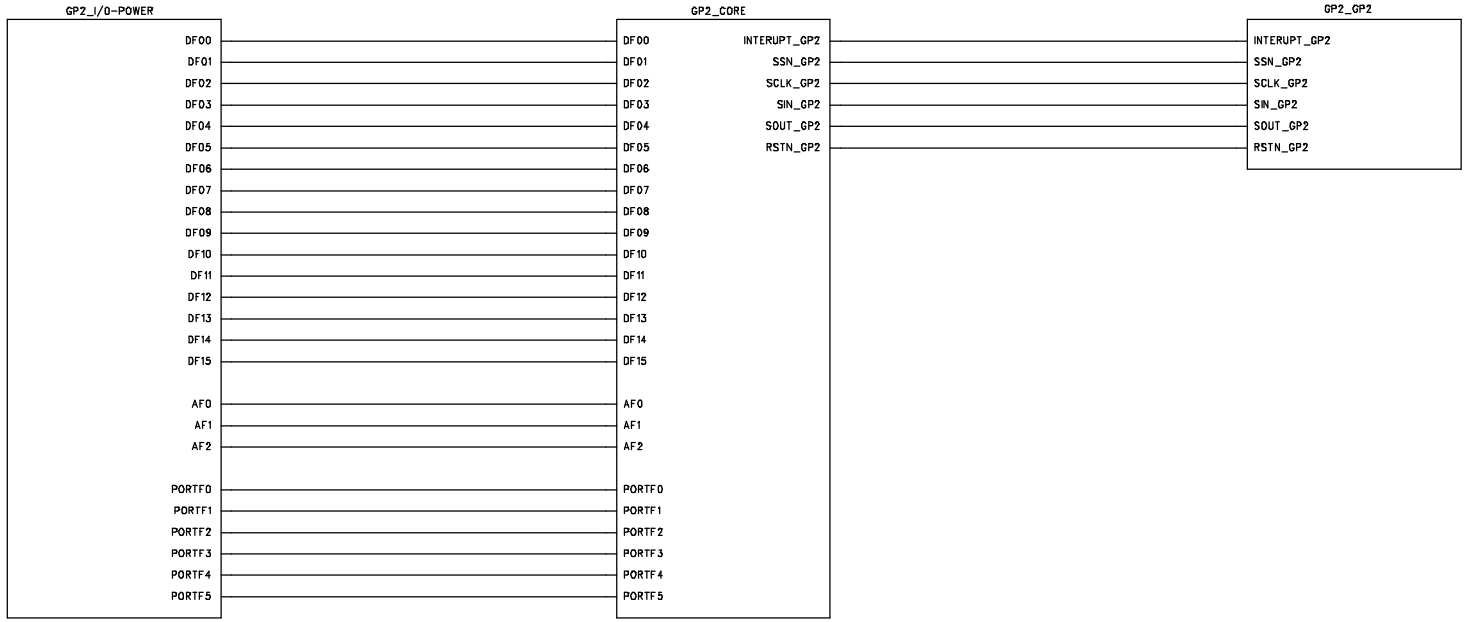
The software collects the displayed measurement data in an array and enables to export them into a file. The “Export data to file “ function stores the measurement values in a *.txt - file format, This data format can be read by several programs, e. g, Microsoft Excel, and enables various possibilities for data postprocessing and visualization.



	Mean 1	Mean 2	Mean 3
1	10031,525		indicator table
2	10031,662		
3	10031,708		
4	10031,654		
5	10031,647		
6	10031,586		
7	10031,708		
8	10031,654		
9	10031,708		
10	10031,647		
11	10031,708		
12	10031,708		
13	10031,601		
14	10031,601		
15	10031,647		
16	10031,708		
17	10031,654		
18	10031,593		
19	10031,670		
20	10031,532		
21	10031,723		
22	10031,586		
23	10031,654		
24	10031,708		
25	10031,715		
26	10031,776		
27	10031,654		

4 AM-GP2 Module

4.1 AM-GP2 Schematics



REVISION RECORD			
LTR	ECO NO:	APPROVED:	DATE:

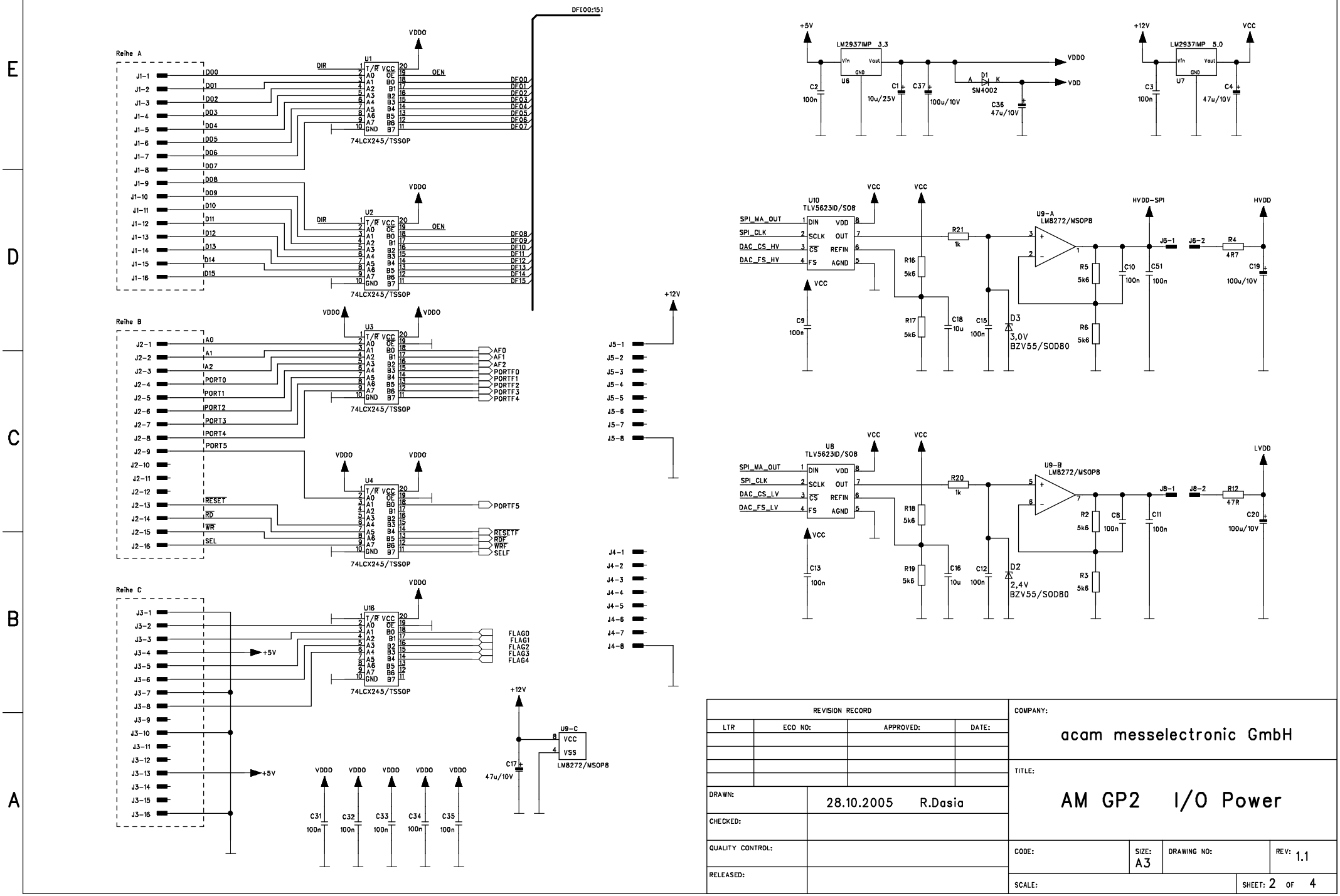
DRAWN:	28.10.2005 R.Dasia
CHECKED:	
QUALITY CONTROL:	
RELEASED:	

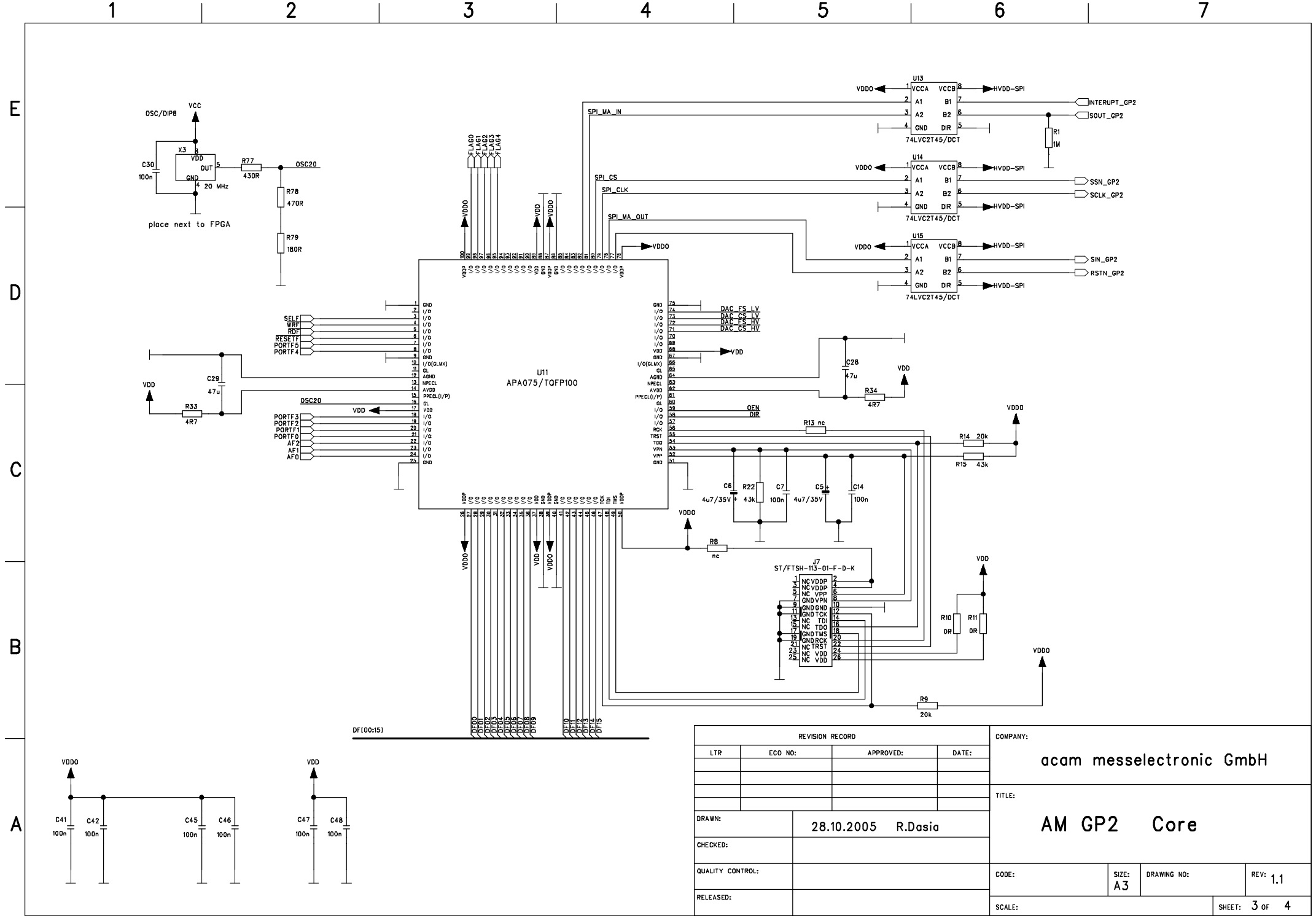
COMPANY: acam messelectronic GmbH

TITLE: AM GP2

CODE:	SIZE: A3	DRAWING NO:	REV:

SCALE: | SHEET: 1 OF 4


Reihe A
Reihe B
Reihe C
DF[00:15]
74LCX245/TSSOP
LM2937IMP 3.3
LM2937IMP 5.0
SM4002
TLV5623ID/SO8
LM8272/MSOP8
3,0V
BZV55/SODB0
2,4V
BZV55/SODB0
28.10.2005 R.Dasia
acam messelectronic GmbH
AM GP2 I/O Power
REVISION RECORD
LTR
ECO NO:
APPROVED:
DATE:
DRAWN:
CHECKED:
QUALITY CONTROL:
RELEASED:
COMPANY:
TITLE:
CODE:
SIZE: A3
DRAWING NO:
REV: 1.1
SCALE:
SHEET: 2 OF 4



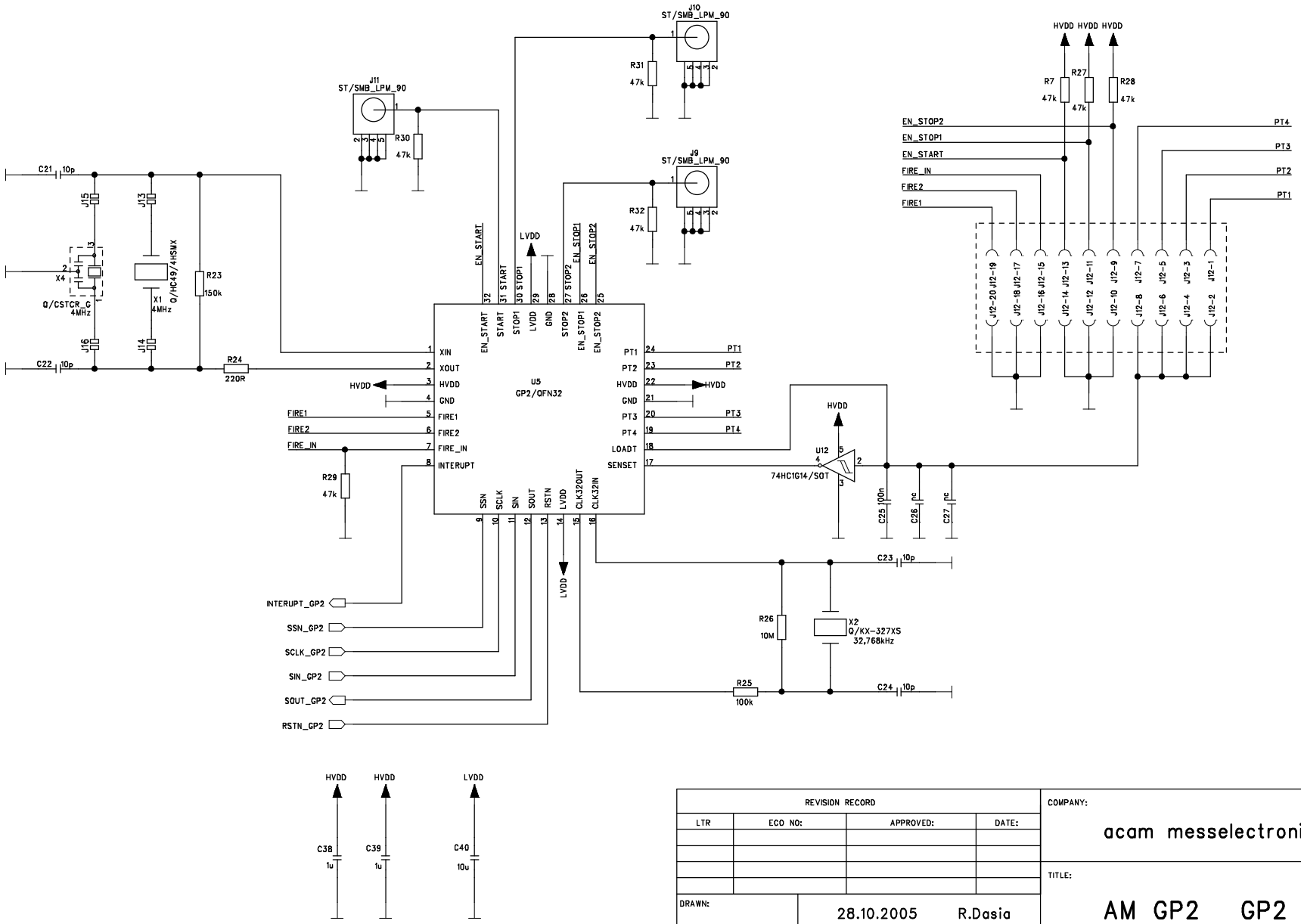
REVISION RECORD			
LTR	ECO NO:	APPROVED:	DATE:

DRAWN:	28.10.2005 R.Dasia
CHECKED:	
QUALITY CONTROL:	
RELEASED:	

COMPANY: acam messelectronic GmbH

TITLE: AM GP2 Core

CODE:	SIZE: A3	DRAWING NO:	REV: 1.1
SCALE:			SHEET: 3 OF 4



REVISION RECORD			
LTR	ECO NO:	APPROVED:	DATE:

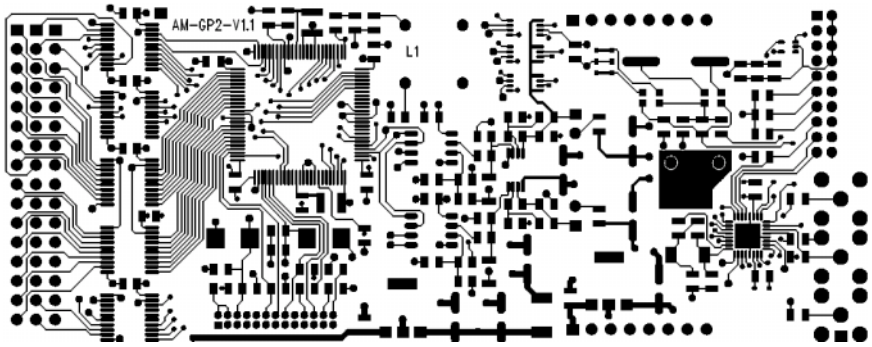
DRAWN:	28.10.2005	R.Dasia
CHECKED:		
QUALITY CONTROL:		
RELEASED:		

COMPANY: acam messelectronic GmbH

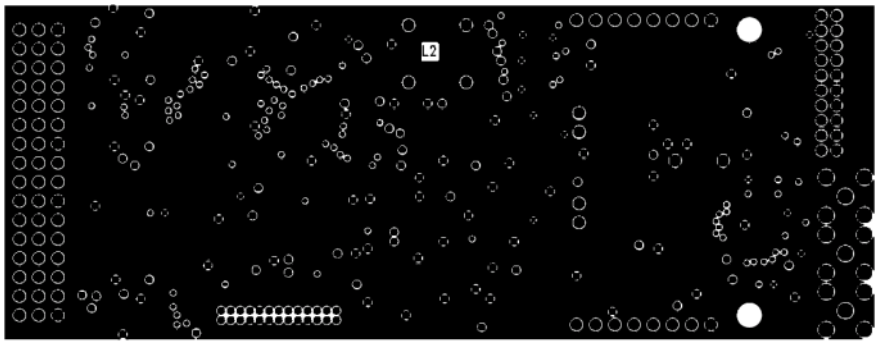
TITLE: AM GP2 GP2

CODE:	SIZE: A3	DRAWING NO:	REV: 1.1
SCALE:			SHEET: 4 OF 4

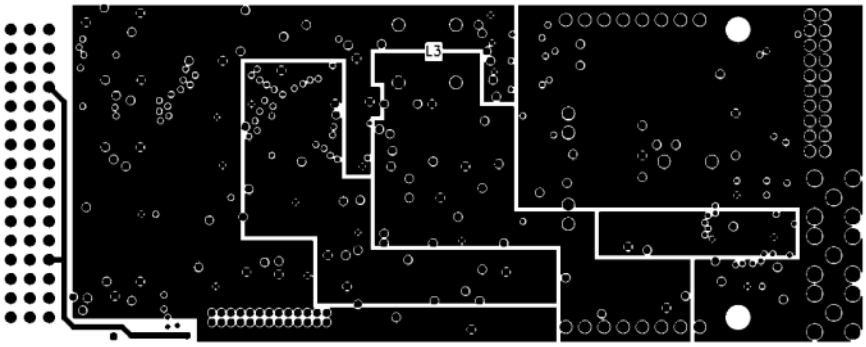
4.2 AM-GP2 Board Layout



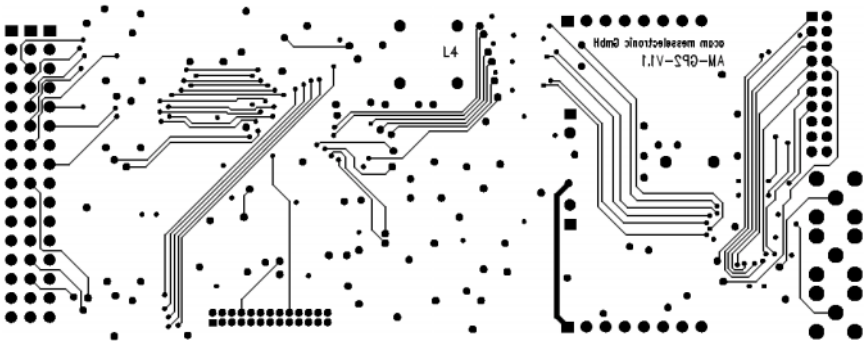
Layer 1



Layer 2, GND

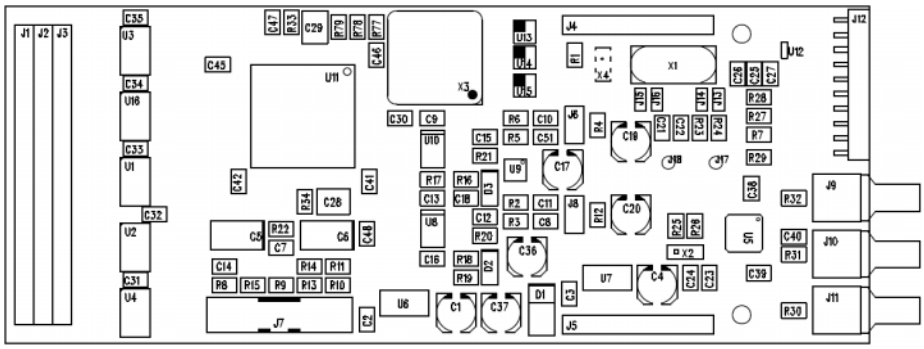


Layer 3, Supply



Layer 4

4.2 AM-GP2 Board Layout



Top, Assembly

Last Changes

First edition: 16th Feb 2006

Contact

Headquarter Germany:	acam-messelectronic gmbh	Am Hasenbiel 27 D - 76297 Stutensee-Blankenloch	Tel: +49-7244-7419-0 Fax: +49-7244-7419-29 email: support@acam.de www.acam.de
Distributors :			
France	microel (CATS S.A.)	Immeuble "Oslo" - Les Fjords 19, avenue de Norvège Z.A. de Courtaboeuf - BP 3 91941 LES ULIS Cedex	Tél. : +33 1 69 07 08 24 Fax : +33 1 69 07 17 23 commercial@microel.fr www.microel.fr
Great Britain	2001 Electronic Components Ltd.	Stevenage Business Park, Pin Green Stevenage, Herts SG1 4S2	Tel. +44 1438 74 2001 Fax +44 1438 74 2001 a.parker@2k1.co.uk www.2k1.co.uk
India	Brilliant Electro-Sys. Pvt. Ltd.	4, Chiplunker Building, 4 Tara Temple Lane, Lamington Road, Bombay – 400 007	Tel: +91 22 2387 5565 Fax: +91 22 2388 7063 www.brilliantelectronics.com besimpex@vsnl.net
Israel	ArazimLtd.	4 Hamelacha St. Lod P.O.Box 4011 Lod 71110	Tel: 972-8-9230555 Fax: 972-8-9230044 email: info@arazim.com www.arazim.co.il
Japan	DMD–Daiei Musen Denki Co., Ltd.	10-10, Sotokanda, 3-Chome, Chiyoda-Ku Tokyo 101-0021	Tel: +81 (0)3 3255 0931 Fax: +81 (0)3 3255 9869 www.daiei-dmd.co.jp sales@daiei-dmd.co.jp
P.R. China	Broadtechs Technology Co. Ltd.	Shanghai Office: 3C JinHuan Building, 489 Xiang Yang Road South Shanghai, 200031	Tel.: +86-21-54654391 Fax: +86-21-64454370 http://www.acam-china.com/ Email: info@acam-china.com
South Korea	SamHwa Technology Co., Ltd.	#4 4F Kyungwon building, 416-6 Jakjeon-dong GYEYANG-GU, INCHEON 407-060	Tel: +82 32 556 5410 Fax: +82 32 556 5411 www.isamhwa.com minjoonho@isamhwa.com
Switzerland	Computer Controls AG	Neunbrunnenstr. 55 8050 Zürich	Tel.: +41-1-308 6666 Fax: +41-1-308 6655 email: roeschger@ccontrols.ch www.ccontrols.ch
United States of America	Transducers Direct, LCC	264 Center Street Miamiville, Ohio 45147	Tel: 513-583-9491 Fax: 513-583-9476 email: sales@acam-usa.com www.acam-usa.com
Russia	Galant Electronics, Ltd.	100, Prospekt Mira, Moscow, 129626, Russia	Tel\Fax: +7-495-987-42-10, Tel: +7-095-107-19-62 Mobile +7-916-993-67-57 Email: leonid-k@galant-e.ru www.galant-e.ru

The information provided herein is believed to be reliable; however, acam assumes no responsibility for inaccuracies or omissions. Acam assumes no responsibility for the use of this information, and all use of such information shall be entirely at the user's own risk. Prices and specifications are subject to change without notice. No patent rights or licenses to any of the circuits described herein are implied or granted to any third party. acam does not authorize or warrant any acam product for use in life support devices and/or systems.