RMAC
Radiation Monitor Alarm Card
Version 2.0

Björn Lundberg,Ulf Mjörnmark
Lund University, Lund, Sweden

Abstract

RMAC is an alarm card in VME. The card contain sixteen counters, these are read by a microcontroller which performs the comparison with the alarm requirements. The status is output to a relay and LED display, and the logarithm of the total rate per second is available as a voltage output. Communication with a remote host is done through FIFOs.

January 15, 1996

Comments to : *jum@quark.lu.se*

This note is available from WWW: *http://www.quark.lu.se/˜jum/documents/rmac.ps*

Contents

List of Figures

List of Tables

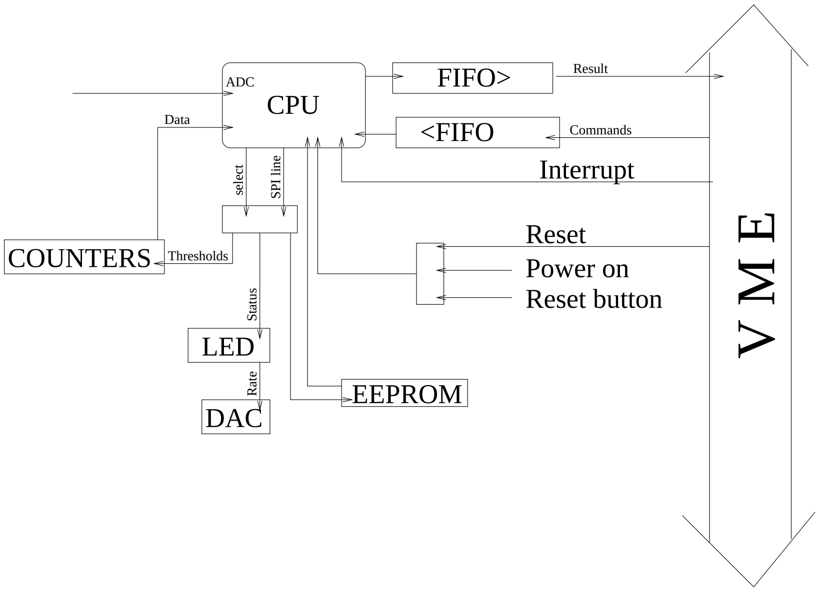

Figure 1: Hardware block diagram

1 Introduction

The RMAC card takes sixteen analog inputs, and counts during a given period the number of pulses above a programmable threshold. A counter is a mixture of hardware and software, a six bit hardware counter is regularly read by a microcontroller. The maximum rate the card can handle is 300 kHz per input. The summed rate of all sixteen inputs is available as a logarithmic voltage. A relay is switched if the rate exceeds a given alarm level during a longer time period.

There are two time periods used. The minor period is the time taken for calculating the rate. The microcontroller adds up the values during one minor period. The major period is the number of minor periods used for deciding on the alarm conditions. The minor period is given by the program running on the microcontroller, while the major period value can be modified by an external host. The minor period is set to one second.

The microcontroller can communicate with an external host through two FIFOs, which are accesable from the VME interface.

Additional eight analog inputs are regularly sampled with an ADC. These inputs are for sampling the temperature given by temperature sensitive resistor devices.

2 Hardware overview

2.1 The microcontroller

The processor used is the PIC16C74 from Microchip [1]. This is a RISC-like, EPROM based 8 bit microcontroller with:

- 200ns instruction cycle
- 33 I/O ports
- 4096 x 14 bit EPROM memory
- 256 x 8 bit SRAM registers
- eight 8 bit multiplexed ADC inputs
- Serial Peripheral Interface - SPI
- Internal watchdog
- External interrupts
- Timers with interrupt

Ports are used to communicate with different peripheral devices. The usage is described in table 1.

EPROM memory contains the program. The lower 2k words are used for the program itself, the upper 2k words for the logarithmic lookup table.

SRAM is a two page memory with 128 bytes each, i.e. a total of 256 bytes. Of these are 64 bytes special registers, 192 bytes are genral purpose registers.

ADC is used to convert the voltage from the eight analog inputs to a digital value. The analog inputs are multplexed to the ADC.

Serial Peripheral Interface - SPI is a serial interface to the serially programmable devices.

Watchdog is an internal timer on the processor, which is set to ca 580 ms. If the watchdog timer is not cleared within this period, the CPU is automatically reset.

External interrupt is connected to the VME bus. An interrupt is generated when the VME address 6 is accessed (read or written).

Timer is used to regularly generate interrupts to the processor. The interval between two interrupts, a tick, is set to 4880 instruction cycles, i.e. 0.976 ms. The 16 bit timer register is loaded with the value (hex)ECF0. One second corresponds to 1024 ticks.

Port	*usage*	*direction*
RA 0-3	Analog input 0-3	IN
4	SPI chain select bit 1	OUT
5	Analog input 4	IN
RB 0	External interrupt	IN
1-4	Counter address	OUT
5	Counter read strobe	OUT
6	FIFO in empty	IN
7	FIFO in read strobe	OUT
RC 0	FIFO out full	IN
1	FIFO out write strobe	OUT
2	SPI chain selecect bit 0	OUT
3	SPI clock	OUT
4	SPI data in	IN
5	SPI data out	OUT
6	Serial transmit	OUT
7	SPI line enable	OUT
RD 0-7	Data bus	IN/OUT
RE 0-2	Analog inputs 5-7	IN

Table 1: Port usage

2.2 Peripherals

The peripherals connected to the processor are:

- EEPROM
- Counter threshold storage
- Counters
- LED display
- Digital to Analog Converter - DAC
- Input/output FIFOs
- Temperatur sensor readout

The EEPROM, Counter thresholds, LED display and DAC are connected to the processor through the SPI-chain. Table 2 gives the conditions for selecting a specific device in the chain.

RC7	*RC2*	*RA4*	*peripheral*
X	0	0	none
1	X	X	none
0	0	1	EEPROM
0	1	0	Counter thresholds
0	1	1	DAC and Display

Table 2: SPI select lines

2.2.1 EEPROM

Parameters are stored in a serial 512x8 EEPROM (X25040). The EEPROM is read/written with 1 Mhz clock rate. Endurance is 100 000 cycles per byte.

Reading the EEPROM is done by selecting the EEPROM in the SPI chain, any number of words can be read as long as it is selected. When all words has been rad, then the selection must be disabled.

Writing the EEPROM is done word by word, i.e. selecting the EEPROM, write the word, deselecting the EEPROM will update the value in the EEPROM location.

2.2.2 Counter threshold storage

The counter thresholds are setup using 2 parallel-loaded octal DACs (AD7228), which gets their serial input data from an 8-bit serial-to-parallel interface implemented in the same programmable chip as the rate counter.

The loading of the threshold DACs is done by writing four words for each counter thresholds, the first word containing the DAC address, the second the threshold value. The third word is the bit pattern , which turns off the addressing of the DAC. The fourth word is a dummy word.

word one XXXWWAAA, where AAA is the DAC address modolus in groups of eight, WW is write enable bits, 00 = illegal, 10 = DAC address 0-7 selected, 01 = DAC addresses 8-15 selected.

word two is the threshold value.

word three XXXWWAAA, with WW=11 disable DAC addressing.

word four dummy word.

The DAC is loaded by selecting the threshold DACs in the SPI chain, this must be done after the four words has been written for each threshold. The DACs must be deslected before the next threshold can be written.

2.2.3 Discriminators

The analog pulses are fed into 16 LEMO connectors which goes to eight double, very fast (4.5 ns) comparator chips (AD9698). These discriminators have added hysteresis to decrease their sensitivity to multiple transition. The 16 outputs from the discriminators are fed to the hardware rate counters.

2.2.4 Counters

The rate counters are implemented in two steps. The reason for doing it this way is to get a sufficiently fast counter (max 300kHz) as well as reducing the workload of the microcontroller. The first step is to use a programmable chip (Lattice ispLSI1032) as a fast hardware rate counter. The configuration of the chip is as eight 6-bit counters with a 1-of-8 output multiplexer. Two of these chips are used to handle all 16 inputs. Once every 0.5 ms, they are read out and reset by the microcontroller. The microcontroller then acts as the second step and adds this value to a register n times during a minor period to give the total rate in Hz.

2.2.5 Digital to Analog(ue) Converter - DAC

The analog rate output should be the logarithm of the rate and is implemented by a look-up table in which the microcontroller converts the frequency to its logarithm and then feeds this value to a double 12-bit serial DAC (MAX532). The reference voltage on this DAC is adjusted to a maximum of 7V, corresponding to a total rate of 10E7, i.e. 10MHz. The maximum resolution of the output is 7/2E12 or 7/4096 which gives a minimum step of 1.7mV. However, in order to decrease the size of the look-up table, not all steps are used. The DAC contains two separate outputs, one is used for the sum of all rate counters and one is used as a single counter selector, which can be changed to any channel through the VME data and is set by default to CH1. The serial chain that controls the data load is physically the same as the chain controlling the LED display and is thus updated each minor period. The value is loaded to the DAC when the select line for DAC and LED display is disabled (see table 2).

2.2.6 LED display

The current status of the card is available on a LED display. It is updated each minor period. The meaning of each LED is given in table 3. The LED display must be written as the first byte in the SPI chain containing the DAC and LED display. The value is loaded to the LED driver when the select line for DAC and LED display is disabled (see table 2).

CPU running is on when the processor has rebooted, it goes off when the processor is reset and stays off until the first minor period has passed.

Heart beat changes its state each minor period.

Connected tells if there is a remote host communicating with the processor through the FIFOs connected to VME. It is off if no host has requested a connection, and on if there is a connection. It stays on until the remote host disconnects.

Rate greater than 16 Hz shows if the summed rate for all inputs are greater than 16 hz during a minor period.

Internal error tells if a internal error occured during the previous minor period. Internally is the type of error stored, and if a remote host is connected

Colour	*meaning*
yellow	CPU running
yellow	Heart beat
yellow	Connected
yellow	Rate greater than 16 Hz
red	Internal error
red	Alarm 0
red	Alarm 1
red	Temperature warning

Table 3: LED display

Relay	*meaning*
1	rate alarm 0
2	rate alarm 1
3	high temperature
4	CPU running

Table 4: Meaning of relays

the error pattern is stored in the FIFO. Table 14 in section 3.8 gives the content of the error pattern.

Alarm 0 and Alarm 1 are the status of the rates, the exact meaning of these alarms are described in the software part of this document.

Temperature warning is set when the input to one of the ADCs exceeds a given value.

2.2.7 Relay outputs

There are four open/close relays which are used to signal the status of the processor and the conditions for high input rates. Table 4 gives the meaning of the realys. An open relay is OK, closed is bad. See section 2.2.6

Alarm 0 and Alarm 1 are the status of the rates, the exact meaning of these alarms are described in the software part of this document.

Temperature warning is set when the input to one of the ADCs exceeds a given value.

CPU running is on when the processor has rebooted, it goes off when the processor is reset and stays off until the first minor period has passed.

2.2.8 Input/output FIFO

The interface to VME consists of two 8kx9 bits DS2013 FIFOs, these are interfaced to the microcontroller data bus, PORTD. They are accessed by setting the

address	*bits*	*description*	*read/write*
0	0-7	FIFO read	read
2	0-7	FIFO write	write
4	0	FIFO read empty (active low)	read
	1	FIFO write full (active low)	read
	2	CPU running (active high)	read
	3	Remote connections (active high)	read
6	0-7	Interrupt CPU	read/write
8	0-7	Reset CPU	read/write

Table 5: VME address map

appropriate read/write strobe. The empty/full flags (active low) are available for microcontroller. Ports used is given in table 1.

The program must first set the read/write strobe, then read/write the data from/to PORTD, and afterward reset the read/write strobe.

2.2.9 Temperature sensor readout

The readout is designed to be used together with the PT1000 temperature sensing resistor. 0 degrees corresponds to 1000 ohm, the resistance increases with 3.1 ohm/degree.

The circuit contains a current source, which sends a current through the temperature sensitive resistor. The potential across the resistor is sampled by the microcontroller ADC, and is adapted to the dynamic rang of the ADC. 0 V corresponds to zero degrees, one degree per 36 mV.

2.3 VME interface

The module is a slave on the VME bus. From the VME is it possible to read/write the FIFOs, reset the card, and issue an interrupt to the microcontroller. The VME base address of the module is set by jumpers on the card. The module is using 24-bit addressing, where the upper four bits are set by the jumpers. Only 16-bit data access is possible, even though the data from the FIFOs is 8-bit. The upper eight bit of the data word is not used. Table 5 gives the VME address map.

The bits for the *CPU running* and *Remote connection*, can be used by a remote host to recognize when the microcontroller has been reset during a connection. *Remote connection* bit is set to zero after a reset.

2.4 Interrupts

Two interrupts are used on the microcontroller,the internal timer interrupt is used to interrupt the microcontroller at 'ticks' intervals. The minor period consist of 1024 ticks,corresponding to one second.

An external interrupt is generated when accessing VME address 6. This interrupt must be used by a remote host to inidcate when a new command is pending in the input FIFO.

3 Software overview

The overall structure of the program is shown in figure 2.

After a reset is the processor registers initialized with their initial settings, and the counters are cleared and their thresholds loaded. At the beginning of a minor period is any pending command executed. After the command execution are the counters read in a loop until the end of the minor period. For each counter are the read values added to the software counter, and to the total sum of all counters. At the end of the minor period is the ADC conversion started, for the next analog input. During the conversion is the alarm handler called, after which the ADC is read. If it is a major period the alarm is cleared/set. If the connect bit is set, the result is stored in the output FIFO, to be read by the remote host. The DAC value that gives the logarithm of the rate, is fetched from a lookup table. This value and the alarm pattern is written to the DAC/LED SPI chain.

In the following are register locations named with capital letters, constants do all start with '_', and compilation flags with '__'.

The usage of the general purpose registers are described in table 6. The addresses are given in hex.

3.1 Buffers

All parameters and data are contained in buffers, with buffer header containing the exclusive buffer length and a buffer identifier. The buffer identifiers are shown in table 7, and contents of each buffer is given in table 8.

3.2 Compilation flags

A number of compilation flags tells the compiler which parts of the code to include in the program.

__ADC includes the ADC readout.

__EEPROM includes EEPROM read/write possibiliteis. Without the EEPROM handling are the program default parameters used after reset, and it is not possible to store/fetch parameters in/from EEPROM.

__FIFOIN includes the input FIFO handling. Without input FIFO are no commands read from the FIFO. All command interrupts are ignored.

__FIFOOUT includes the FIFO output handling. Without the output FIFO are no results or status written to the output FIFO.

__ADAC includes ALARM and DAC handling. Without this flag will nothing be displayed on the LED display and no level on the logarithmic rate outputs.

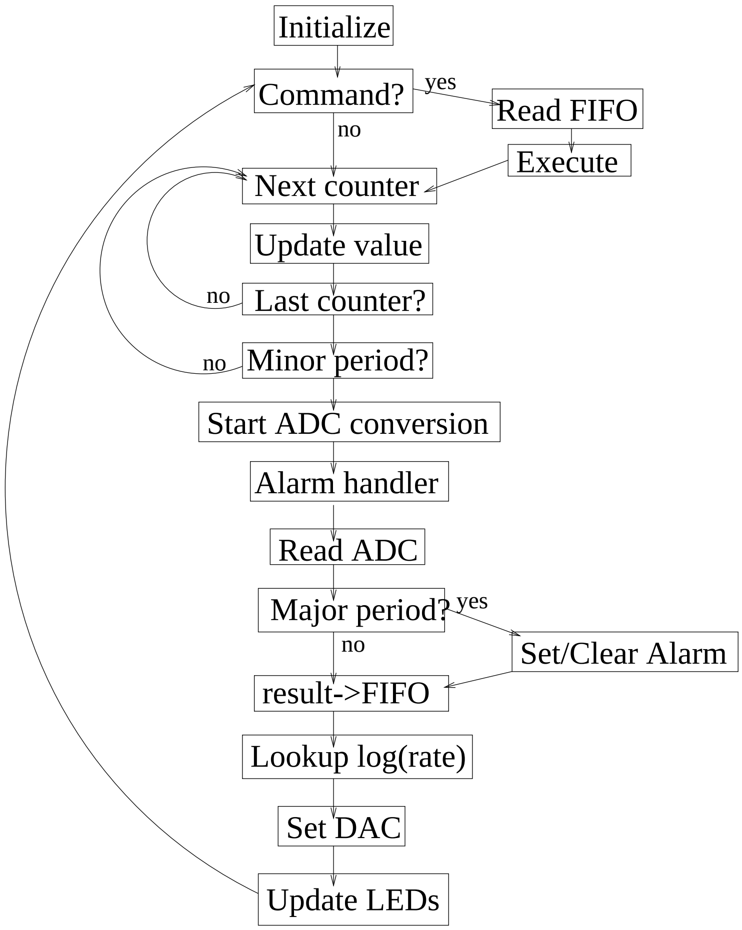

Figure 2: Software block diagram

Page 0		
Address	*length*	*usage*
CNT0	_NBCNT*3	counters, counters have byte order 321
RES0	_NBRES	result buffer
ERR0	_NBERR	alarm, error flag
PAR0	_NBPAR	parameters
DAC0	_NBDAC	DAC values for voltage output
VAR0_B0	_NBVAR_B0	variables
7E(WBUF)		interrupt storage of W register
7F(LOOPCNT)		loop counter
Page 1		
Address	*length*	*usage*
THR0	_NBTHR	Thresholds for the counters
DATE0	_NBDATE	Date and time of EEPROM parameters
PRG0	_NBPRG	Program version
CMD0	_NBCMD	Command information
LOGTEST0	_NBLT	Logic test parameters
VAR0_B1	_NBVAR_B1	variables
8E(WBUF)		interrupt storage of W register
8F(LOOPCNT)		loop counter

Table 6: SRAM usage

length	Identifier	Value	Register address	Description
_NPAR+1	_UPDPAR	0	PAR0	Parameters
_NBTHR	_UPDTHR	1	THR0	Counter thresholds
_NBDATE+1	_DATEINFO	2	DATE0	Date of EEPROM storage
_NBRES+1	_RESINFO	3	RES0	Results
_NBPRG+1	_UPDPRG	4	PRG0	Program version
_NBCMD+1	_CMDINFO	5	CMD0	Command
_NBLT+1	_TESTINFO	6	LOGTEST0	Logic test parameters
_NBERR+1	_ERRINFO	7	ERR0	Error/status
_NBDAC+1	_DACINFO	8	DAC0	DAC settings

Table 7: Buffer identifiers

_UPDPAR		
Word	*Menomic*	*Content*
0	AD_8_15	Bit pattern of active diodes, number 8 to 15
1	AD_0_7	Bit pattern of active diodes, number 0 to 7)
2	AT_0_7	Bit pattern of active temperature sensors
3	ALPER0	Alarm 0 periods
4	ALPER1	Alarm 1 periods
5	DAT0	Diodes above threshold alarm 0
6	DAT1	Diodes above threshold alarm 1
7	NBSEC0	Number of seconds above threshold alarm 0
8	NBSEC1	Number of seconds above threshold alarm 1
9	THR03	byte 3 of alarm 0 threshold
10	THR02	byte 2
11	THR01	byte 1
12	THR13	byte 3 of alarm 1 threshold
13	THR12	byte 2
14	THR11	byte 1
15	TEMPW	temperature warning in ADC counts
16	SELDIOD	selected diod for rate output
_UPDTHR		
Word	*Menomic*	*Content*
0-15	THR00-THR15	Counter thresholds counter 0-15
_DATEINFO		
Word	*Menomic*	*Content*
0	PARSET	Parameter set to fetch, 07
1	DAY	Day when parameters where stored
2	MONTH	Month when parameters where stored
3	YEAR	Year when parameters where stored
4	HOUR	Hour when parameters where stored
5	MINUTE	Minute when parameters where stored
6	WRCYC3	Number of EEPROM write cycles byte 3
7	WRCYC2	Number of EEPROM write cycles byte 2
8	WRCYC1	Number of EEPROM write cycles byte 1

_RESINFO		
Word	*Menomic*	*Content*
0	SUMCNT3	Summed rate of all counters byte 3
1	SUMCNT2	Summed rate of all counters byte 2
2	SUMCNT1	Summed rate of all counters byte 1
3	SELCNT3	Summed rate of selected counter byte 3
4	SELCNT2	Summed rate of selected counter byte 2
5	SELCNT1	Summed rate of selected counter byte 1
6	DF_8_15	Counters 8 to 15 above threshold
7	DF_0_7	Counters 0 to 7 above threshold
8	ADCVAL	ADC value read
9	SECONDS	Seconds passed
10	ALARM	Alarm/status pattern
11	ERRFLAG	Error flags
_UPDPRG		
Word	*Menomic*	*Content*
0	VERSION	Program version
1	REVISION	Program revision
_CMDINFO		
Word	*Menomic*	*Content*
0	CMDSTAT	Command status
1	CMDNB	Command number
2	CMDDATA	First command data word
_TESTINFO		
Word	*Menomic*	*Content*
0	LTLOOP	Current value
1	LTDIV	Loop divider
2	LTINCR	Simulated counter value
3	LTLOOP2	Current loop number, byte 2
4	LTLOOP1	Current loop number, byte 1
5	LTAD_8_15	Counters to use, 8 to 15
6	LTAD_0_7	Counters to use, 0 to 7
_ERRINFO		
Word	*Menomic*	*Content*
0	ALARM	Alarm/status pattern
1	ERRFLAG	Error flags
_DACINFO		
Word	*Menomic*	*Content*
0	DAC00	DAC value output 0 byte 0
1	DAC01	DAC value output 0 byte 1
2	DAC10	DAC value output 1 byte 0
3	DAC11	DAC value output 1 byte 1

Table 8: Buffer contents

__THRDAC includes the setting of the thresholds for the counters. Without this flag will the counter thresholds be undefined.

__INTR includes the interrupt handling, i.e. the command and timer interrupts.

__REMOTE is used to force the writing of results to the output FIFO, irrespective if there is any remote connection or not. Without this flag is the result always written to the output FIFO,and after a reset is the program version written to the FIFO.

__SCI includes the serial communication interface. With this flag is a boot message written to an attached terminal.

__MPSIM selects code specific for debugging on the software simulator.

__DEBUG1 sets the software counters to an initial value.

__DEBUG2 performs the LED display test at reboot.

__LOGICDEBUG forces the logic test to be used.

3.3 Initialization

3.3.1 Port initialization

The initial port initialization values are given in table 9 for ports in register page 0 and 10 for ports in register page 1. If a value depends on a compilation flag, is this enclosed within the compilation reguirement. The ports appear in alphabetic order. The meaning of the ports and bit pattern are described in [1]

3.3.2 Default parameters

The EEPROM address space is divided into eight areas, with 64 bytes each. These areas are used to store different parameter sets, 0 to 7. The parameter set at address 0 is fetched at startup. Table 11 gives the parameters stored for each parameter set.

The EEPROM is first looked at address 0. If it does not contains valid buffer identifiers for date and program buffer, then it is uses program default settings for all parameters.

The program contains default settings for all parameters. These are used after a reboot if the EEPROM is not accessible or do not contain any parameter values.

Table 12 gives for each parameter its name and default setting.

3.4 Counter readout

The sixteen counters are read in a loop, the read value is added to the software counter. Before the readout loop starts is the watchdog timer reset. The counters are read until the minor time period has elapsed.

A counter address is set up in bits 1-4 in PORTB, the data is available on the data bus (PORTD) when the counter strobe is set.

If the logic test is enabled, LTLOOP is initialized to the value given by LTDIV, and decremented with one for each readout loop. When LTLOOP

Port	*constant*	*bit pattern*
ADCON0	_ADCON0	10000001
PORTA	_INITA	00000000
PORTB	_INITB	10000000
PORTC	_INITC	10000010
PORTD	_INITD	00000000
PORTE	_INITE	00000000
RCSTA	_RCSTA	00010000
SSPCON	_SSPCON	00110000
TMR1L	_TTICKL	11110000
ifdef __MPSIM		
TMR1H	_TTICKH	11111111
else		
TMR1H	_TTICKH	11101100
endif		

Table 9: Port initialization - page 0

Port	*constant*	*bit pattern*
ADCON1	_ADCON1	00000000
ifdef __MPSIM		
OPTION_REG	_OPTION	11111000
else		
OPTION_REG	_OPTION	11111001
endif		
PCON	_PCON	00000010
SPBRG	_SPBRG	00000100
TRISA	_PORTA	00101111
TRISB	_PORTB	01000001
TRISC	_PORTC	00010001
TRISD	_PORTD	11111111
TRISE	_PORTE	00000111
TXSTA	_TXSTA	00100100

Table 10: Port initialization - page 1

Identifier	*entries*	*content*
_DATEINFO	_NBDATA + 1	Date of storage,cycles in EEPROM
_UPDPRG	_NBPRG + 1	Progam version,revision
_UPDPAR	_NBPAR + 1	Program parameters
_UPDTHR	_NBTHR + 1	Thresholds

Table 11: EEPROM parameter storage

_UPDPAR		
name	*description*	*program value*
AD_8_15	Bit pattern of active diodes, number 8 to 15	FF(hex)
AD_0_7	Bit pattern of active diodes, number 0 to 7	FF(hex)
AT_0_7	Bit pattern of active temperature sensors	FF(hex)
ALPER0	Alarm 0 periods	60 seconds
ALPER1	Alarm 1 periods	60 seconds
DAT0	Diodes above threshold alarm 0	4
DAT1	Diodes above threshold alarm 1	4
NBSEC0	Number of seconds above threshold alarm 0	60
NBSEC1	Number of seconds above threshold alarm 1	60
THR03	byte 3 of alarm 0 threshold = 10 kHz	00 (hex)
THR02	byte 2	27 (hex)
THR01	byte 1	10 (hex)
THR13	byte 3 of alarm 1 threshold = 1 kHz	00 (hex)
THR12	byte 2	27 (hex)
THR11	byte 1	10 (hex)
TEMPW	temperature warning in ADC counts	7F (hex)
SELDIOD	selected diod for rate output	0
_UPDTHR		
name	*description*	*program value*
THR00-THR15	Counter thresholds	10

Table 12: Program parameter defaults

__ALARM_HANDLER	*Algorithm*
0	Fixed interval

Table 13: Alarm handler algorithms

reaches zero, the counter value read is taken from LTINCR, else set to zero. The current loop number, LTLOOP1 and LTLOOP2, is incremented with one and LTLOOP reinitialized to LTDIV.

3.5 ADC readout

After the minor period is the ADC conversion started, unless the previous conversion has not finished. The end of conversion is checked after the alarm handling, and the ADC is read if the conversion has finished. If the conversion has not finished then the ADC timeout bit is set in the error flag register.

The temperature warning is set if the read value (ADCVAL) is greater then TEMPW, and the sensors is marked as active. Only the upper 5 bits of the ADC value is stored in ADCVAL, the lower 3 bits of ADCVAL is the sensor number.

3.6 Alarm handler algorithms

The alarm handler algorithm is implemented in software, this makes it possible to modify it. This is one reason to have a microcontroller on the card. The currently implemented alarm handlers are given in table 13. The selection of the alarm handler is done at compilation time by setting the compilation flag __ALARM_HANDLER to one of the values given in table

The requirement for an alarm is that at least DAT out of the 16 inputs have a rate higher than a threshold - THR, during NBSEC out of ALPER minor periods. The rate calculation is described for each algorithm below.

3.6.1 Fixed interval algorithm

The algorithm (figure 3) add up the rate for all minor periods during the major period. This value is compared with preset thresholds, and an alarm is issued if the total rate during the major period exceeds the threshold. The major period is fixed in time, i.e. for an alarm to come active must the rate be high for at least the major period interval, and up to two major periods.

3.7 Lookup tables

The rate is output as a logarithmic voltage by writing a value to the 12-bit DAC. It takes to long time to convert the rate to a logarithmic value in software, this conversion is instead done with a lookup table. This lookup table is generated on a computer by a program written in C. Since there is limited space on the

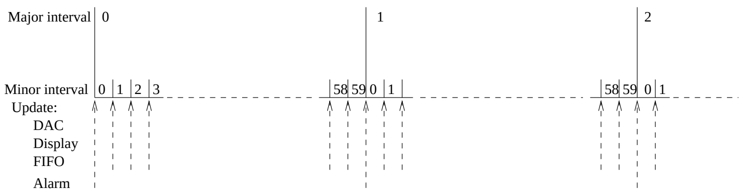

Figure 3: The fixed interval algorithm

Error bit	*meaning*
0	There was watchdog reset
1	An ADC timeour occured
2	Error in FIFO read transfer
3	Error in EEPROM transfer
4	Error in command
5	Error in FIFO write transfer
6	NOT DEFINED
7	NOT DEFINED

Table 14: Internal errors

microcontroller, 2k, are not all possible rates included in the table. For small rates (0-16) are there no missing entries, but the number of missing entries increases with the rate. For very large rates, MHz, are the gaps 64k. Due to the limited resolution of the DAC, and the logarithmic behaviour will this barely be visible. For rates inbetween two entries, is the lower value used for the DAC setting.

3.8 Error handler

Several errors are recognised by the microcontroller program (table 14). The watchdog bit is set when the microcontoller has been reset by the watchdog timer. FIFO transfer errors are set when the FIFOs are flagged as full, or unexpectedly empty. An EEPROM error occur when there is no recognisable data at the requested address. If the command identifier or command data is not understood, a command error is flagged.

All errors except the ADC timeout sets the error bit in the status word displayed on the LED. The errors stays on for one minor period, after this time they are cleared.

3.9 External communication

Through the FIFOs is it possible to communicate with the microcontroller program. The information exchanged are commands to the microcontroller pro-

Buffer content		
word	*mnemonic*	*content*
0	_NBCMD	Length of command buffer (exclusive)
1	_CMDINFO	Buffer identifier
2	CMDLEN	Command length (exclusive)
Command buffer		
word	*mnemonic*	*content*
0	CMDSTAT	Command status
1	CMDNB	Command number
2	CMDDATA	First command data word

Table 15: Command buffer content

gram and results/status from the microcontroller. To enable the writing of the rsults/status must the command *connect* be sent to the microcontroller. To disable the writing, send *disconnect*. Commands can be sent at any time. For the microntroller to read the input FIFO must the remote host generate an interrupt to the controller. This is done by writing into the interrupt address (see table 5). The reading of the FIFO is done at the beginning of each minor period.

3.9.1 Commands

The commands must be preceded by the buffer header containing the total length of the command buffer, excluding the length itself, the command identifier, and the exclusive length of the following command buffer. The format is given in table 15.

Below is a list of the recognised commands, and the meaning of them. The details of the individual data words for each command are given in table 16-17. Those words marked '(Read)' are only used to send information from the RMAC program. Any values set by the remote host is ignored.

When the command has been performed is the (updated) command buffer written to the output FIFO.

0: _CMD_AD changes the status of active diodes.

1: _CMD_AT changes the status of active temperature sensors.

2: _CMD_ALPER sets the alarm periods.

3: _CMD_DAT sets the number of diodes required to be above the thresholds.

4: _CMD_NBSEC sets number of seconds during an alarm period for which the alarm conditions must be fullfilled.

5: _CMD_THR sets the alarm thresholds.

7: _CMD_SD sets the selected diod for rate monitoring.

8: _CMD_CTHR sets all the counter thresholds.

9: _CMD_CON connect/disconnect external communication.

10: _CMD_FE fetch a parameter set from EEPROM.

11: _CMD_SE store the current parameters as a parameter set in EEPROM.

12: _CMD_FF fetch parameters from FIFO, the parameters are contained in the following buffer. This command do not contain any data words

13: _CMD_SF store current parameters in FIFO. This command do not contain any data words.

14: _CMD_FD fetch program default parameters. This command do not contain any data words.

15: _CMD_DT perform the display test. This command do not contain any data words.

16: _CMD_DACT perform the DAC test.

17: _CMD_LT perform program logic test.

3.9.2 Status and results

For each minor period writes the RMAC program the result to the output FIFO. This is done if there is a remote connection established. The content of the result buffer is given in table 18. The result stored is the sum of the individual counters, and the rate for the selected counter. For each counter is it indicated if the the counter has exceeded any rate threshold. This is stored in a bit pattern with one bit for each counter, a bit set means the counter has exceeded at least one of the thresholds. The buffer ends with the status/alarm (table 3) and error flags (table 14.

_CMD_AD		
Word	*Mnemonic*	*Content*
0	AD_8_15	Active diodes
1	AD_0_7	Active diodes
_CMD_AT		
Word	*Mnemonic*	*Content*
0	AT_0_7	Active temperature sensors
_CMD_ALPER		
Word	*Mnemonic*	*Content*
0	ALPER0	Period for alarm 0
1	ALPER1	Period for alarm 1
_CMD_DAT		
Word	*Mnemonic*	*Content*
0	DAT0	Diodes required for alarm 0
1	DAT1	Diodes required for alarm 1
_CMD_NBSEC		
Word	*Mnemonic*	*Content*
0	NBSEC0	Seconds required period 0
1	NBSEC1	Seconds required period 1
_CMD_THR		
Word	*Mnemonic*	*Content*
0	THR03	Byte 3 for alarm 0 threshold
1	THR02	Byte 2 for alarm 0 threshold
2	THR01	Byte 1 for alarm 0 threshold
3	THR13	Byte 3 for alarm 1 threshold
4	THR12	Byte 2 for alarm 1 threshold
5	THR11	Byte 1 for alarm 1 threshold
_CMD_TEMPW		
Word	*Mnemonic*	*Content*
0	TEMPW	Temperature warning level
_CMD_SD		
Word	*Mnemonic*	*Content*
0	SELDIOD	Selected diod for rate
_CMD_CTHR		
Word	*Mnemonic*	*Content*
0-15		Counter thresholds for diodes 0-15

Table 16: Parameter update commands

_CMD_CONNECT		
Word	*Mnemonic*	*Content*
0	CONNECT	0/1 : close/open connection,
	_CMD_FE	
Word	*Mnemonic*	*Content*
0	PARSET	Parameter set to fetch, 07
1	DAY	Day when parameters where stored (Read)
2	MONTH	Month when parameters where stored (Read)
3	YEAR	Year when parameters where stored (Read)
4	HOUR	Hour when parameters where stored (Read)
5	MINUTE	Minute when parameters where stored (Read)
6	WRCYC3	Number of EEPROM write byte 3 (Read)
7	WRCYC2	Number of EEPROM write byte 2 (Read)
8	WRCYC1	Number of EEPROM write byte 1 (Read)
	_CMD_SE	
Word	*Mnemonic*	*Content*
0	PARSET	Parameter set to store, 0-7
1	DAY	Day
2	MONTH	Month
3	YEAR	Year
4	HOUR	Hour
5	MINUTE	Minute
6	WRCYC3	Number of EEPROM write byte 3 (Read)
7	WRCYC2	Number of EEPROM write byte 2 (Read)
8	WRCYC1	Number of EEPROM write byte 1 (Read)
	_CMD_DACT	
Word	*Mnemonic*	*Content*
0	DTLOOP	Number of values
1	DTSUM3	First value byte 3
2	DTSUM2	First value byte 2
3	DTSUM1	First value byte 1
4	DTCNT3	Increment byte 3
5	DTCNT2	Increment byte 2
6	DTCNT1	Increment byte 1
	_CMD_LT	
Word	*Mnemonic*	*Content*
0	LTLOOP	Current value
1	LTDIV	Loop divider
2	LTINCR	Simulated counter value
3	LTLOOP2	Current loop number, byte 2
4	LTLOOP1	Current loop number, byte 1
5	LTAD_8_15	Counters to use, 8 to 15
6	LTAD_0_7	Counters to use, 0 to 7

Table 17: Miscelleanous commands

word	*mnemonic*	*content*
0	_NBRES + 1	Total exclusive length
1	_UPDPAR	Buffer identifier
2	SUMCNT3	Summed rate of all counters byte 3
3	SUMCNT2	Summed rate of all counters byte 2
4	SUMCNT1	Summed rate of all counters byte 1
5	SELCNT3	Summed rate of selected counter byte 3
6	SELCNT2	Summed rate of selected counter byte 2
7	SELCNT1	Summed rate of selected counter byte 1
8	DF_8_15	Counters 8 to 15 above threshold
9	DF_0_7	Counters 0 to 7 above threshold
10	ADCVAL	ADC value read
11	SECONDS	Seconds passed
12	ALARM	Alarm/status pattern
13	ERRFLAG	Error flags

Table 18: Result buffer content

References

[1] Microchip data book

A Version history

Major changes is indicated by a new version number, small changes and important bug fixes changes the revision number.

The software for different versions are selected by setting _VERSION to the proper version number.

A.1 version 1.0 - in use from 10 october 1995

This is the original version, without software support for the EEPROM and external communication.

A.2 version 1.1 - tested, not in use

The bit patterns, DF_0_7 and DF_8_15, is added the the result buffer. The bit patterns tells which counters that had a rate above threshold during the last minor period.

A.3 version 2.0 -not tested

The sofwtare support for EEPROM and external communication fully implemented. The display, DAC and logic tests added.