Chapter 2

SYSTEM

Edition 3

Contents

OVERVIEW

This class of Function Blocks provide an interface between the user's application program and the PC3000 Real Time Operating System. Blocks which provide access to parameters which control the system start up, access to the Real Time Clock and facilities for display of diagnostic messages are included.

Refer to the Real Time Operating System Reference for further information on the use of these Function Blocks.

PcsSTATE FUNCTION BLOCK

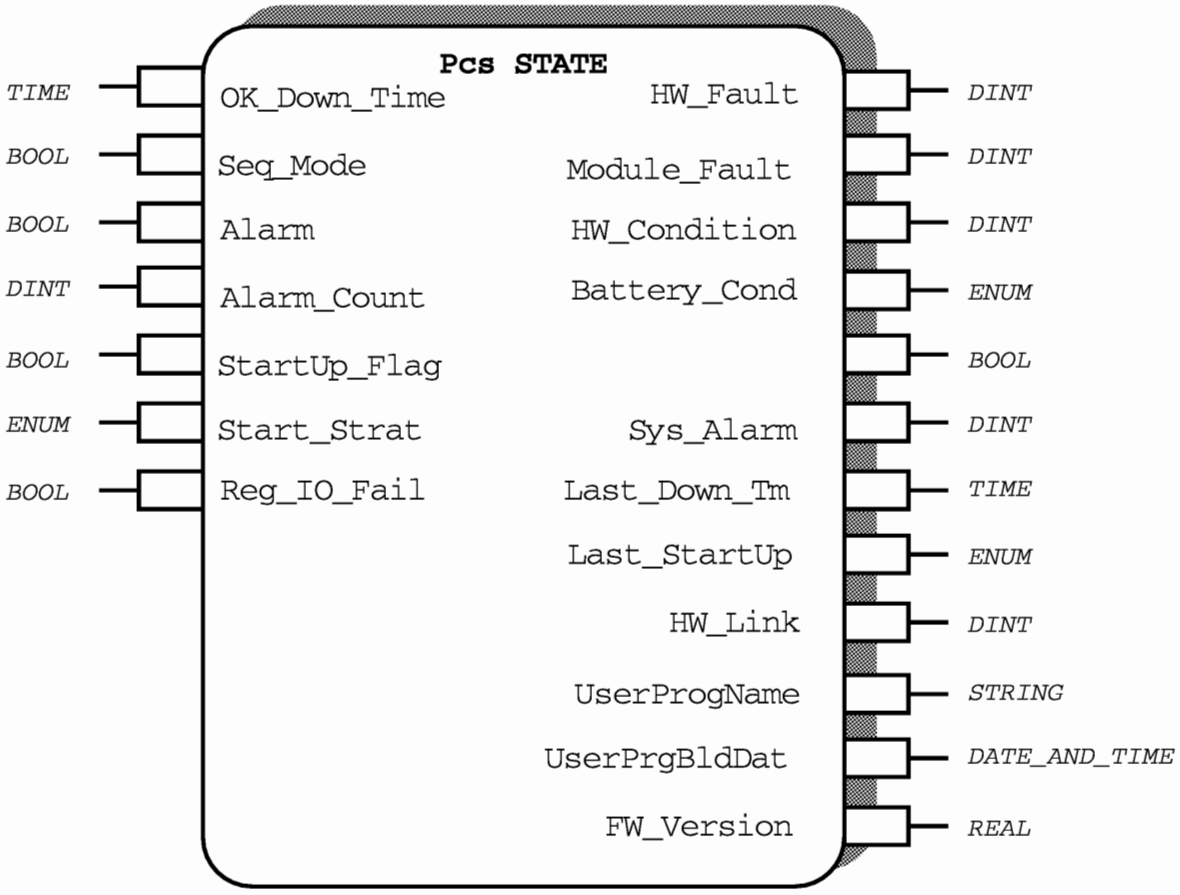


Figure 2-1 PcsState Function Block Diagram

Functional Description

The PcsSTATE function block provides an interface to PC 3000 real time system. All user programs have an instance of this block automatically. There is no need for the user to instantiate one.

Function Block Attributes

Type:..... 8 10

Class: ..... SYSTEM

Category: ..... system

Default Task: ..... Task_2

Short List: ..... Alarm, Alarm_Count, Sys_Alarms

Parameter Descriptions

OK_Down_Time (ODT)

OK_Down_Time defines the maximum acceptable duration of a power failure that can be accepted by the system and still be warm started. This time must include the time that it takes the system to run the start-up diagnostics following the restart (approximately 3 sec). **OK_Down_Time** is only accessed by the system when the start-up strategy parameter **Start_Strat** is set to W_D_E_C (4) or W_D_E_N (5).

Seq_Mode (SM)

Seq_Mode controls the mode of the sequence program. If **Seq_Mode** is set to Hold (1) the sequence program execution is suspended, i.e. active steps and transitions are not executed. If **Seq_Mode** is set to Normal (0) the sequence program is able to run normally.

Note: This parameter has no function in system firmware releases 2.09 and 2.27

Alarm (AL)

The parameter **Alarm** has no function. It has been included for future enhancements.

Alarm_Count (ALC)

Alarm_Count is displayed at the top right corner of the PC3000 Programming Station screen. If **Alarm_Count** is zero, the display reads"OK". It is intended to be set in the user program, either through wiring or in ST steps.

StartUp_Flag (STF)

Start Up_Flag can be used to detect whether the PC3000 LCM has been powered down. It is set to On (1) by the system if the LCM is powered down while a user program is running. It can therefore be used by the user program to detect whether the system has been powered down and restarted.

Start_Strat (SUS)

Start_Strat specifies the start up strategy to be used by the system. It has six possible values:

Do_Not (0):..... The user program will be prevented from starting up.

Cold (1):..... The user program will be Cold started.

Warm (2): ..... The user program will be Warm started.

W_Els_C (3): ..... Warm start, but if that fails (Else) Cold start.

W_D_E_C (4):..... Warm start if the system is within **OK_Down_Time,** else Cold start.

W_D_E_N (5): .... Warm start if the system is within **OK_Down_Time**, else Do Not start up.

Reg_IO_Fail (RIF)

When **Reg_IO_Fail** is set to Yes (1), all errors detected while communicating with the I/O Modules are recorded in the system error log.

HW_Fault (HWF)

HW_Fault has no function. It has been included for future enhancement.

Module_Fault (MF)

Module_Fault provides an indication of the failure of a hardware module. If **Module_Fault** is 0, then no module failures have been detected. If Module_Fault is non zero, the number indicates which module has failed (or the first module if more than one has failed), with digits split into fields R SS CC, where R is the rack number, SS is the slot number and CC is the channel number in the module.

E.g. 30402 indicates that there is a failure in rack 3, module 4, channel 2.

HW_Condition (HWC)

HW_Condition has no function. It has been included for Future enhancement.

Battery_Cond (BAC)

Battery_Cond reflects the condition of the LCM RAM battery. It can have three possible values:

Good (0):..... The battery is healthy.

Low (1): ..... The battery voltage is low and the battery should be replaced.

Faulty (2):..... The battery voltage is insufficient to maintain the memory.

Sys_Alarms (SAC)

Sys_Alarms contains a count of the number of system alarms that have occured. The system alarm details are contained in the error log.

Last_Down_Tm (LDT)

Last_Down_Tm records the duration that the system was down since the last shutdown. This includes the time taken to run the power up diagnostics when the system is powered up. **Last_Down_Tm** will only record the power down time

when the **Start_Start** is set to W_D_E_C(4) or W_D_E_N(5). The parameterwill record time from 0 to the value of **OK_Down_Time**. If the power down period exceeds this the time will be registered as zero.

Last_Start_Up (LSU)

Last_Start_Up indicates the type of the last start up that was performed. It can have one of four values:

None (0): .....User program has not been started up.

Cold (1): .....The User program has been cold started.

Warm (2): .....The user program has been warm started.

Cold_WF (3): ..The user program has been cold started after a failed warm start.

HW_Links (LKS)

HW_Links indicates the setting of the hardware links on the LCM, which are used to set the Group Indentifier for the Eurotherm Bisync communications. This is set by links on VERSION 1 LCM's and by a rotary switch on VERSION 2 Modules. In either case, **HW_Lnks** reflects the current setting.

UserProgName (UPN) [Version 3.00 onwards]

UserProgName indicates the name of the user program (application program) which is currently running in the LCM.

UsrPrgBldDat (BDT) [Version 3.00 onwards]

UsrPrgBldDat indicates the time and date that the user program shown by UserProgName was built. This parameter can be used to determine the version of the user program currently running.

FW_Version (FWV) [Version 3.00 onwards]

FW_Version indicates the version number of the LCM system firmware.

Parameter Attributes

Name	Type	Cold Start	Read Access	Write Access	Type Specific Information	
Alarm	**BOOL**	No(0)	Oper	Super	Senses	No(0) Yes (1)
Alarm_Count	**DINT**	0	Oper	Super	High Limit Low Limit	999 0
Battery_Cond	**ENUM**		Oper		Senses	Good (0) Low (1) Faulty (2)
HW_Condition	**DINT**	0	Oper		High Limit Low Limit	10,000 0
HW_Fault	**DINT**	0	Oper		High Limit Low Limit	10,000 0
HW_Links	**DINT**		Oper		High Limit Low Limit	1000 0
Last_Down_Tm	**TIME**	0	Oper			
Last_Start_Up	**ENUM**	Cold (0)	Oper		Senses	None (0) Cold (1) Warm (2) Cold WF (3)
Module_Fault	**DINT**	0	Oper		High Limit Low Limit	10,000 0
OK_Down_Time	**TIME**	0	Oper	Super		
Reg_IO_Fail	**BOOL**	No(0)	Config	Config	Senses	No (0) Yes (1)
Seq_Mode	**BOOL**	Normal (0)	Oper	Super	Senses	Normal (0) Hold (1)
StartUp_Flag	**BOOL**	Off (0)	Oper	Super	Senses	Off (0) On (1)
Sys_Alarms	**DINT**	0	Oper		High Limit Low Limit	10,000 0

Table 2-1 PcsState Parameter Attributes

Name	Type	Cold Start	Read Access	Write Access	Type Specific Information	
UserProgName	**STRING**	"	Oper	Block	Max String Length	8 Characters
UsrPrgBldDat	**DATE_AND_TIME**	Jan 1 1970 00:00:00	Oper	Block	High Limit Low Limit	Jan 19 2038 03:14:06 Jan 1 1970 00:00:00
FW_Version	**REAL**		Oper	Block		

Table 2-1 PcsState Parameter Attributes (continued)

MESSAGES FUNCTION BLOCK

Figure 2-2 Message Function Block Diagram

Functional Description

The Messages function block enables two strings to be displayed in the message display line at the top of all On-Line PC3000 programming station screens. The messages can have up to 40 characters each and are displayed on the left and right sides of the message display line. All user programs have an instance of this block created automatically.

Function Block Attributes

Type:..... 8 20

Class: .....SYSTEM

Default Task: .....Task_2

Instance Memory Requirement: .....84 Bytes

Parameter Descriptions

P_Message (PM)

The parameter **P_Message** holds the primary message to be displayed on the left half of the diplay line. The input string can be up to 40 characters long.

S_Message (SM)

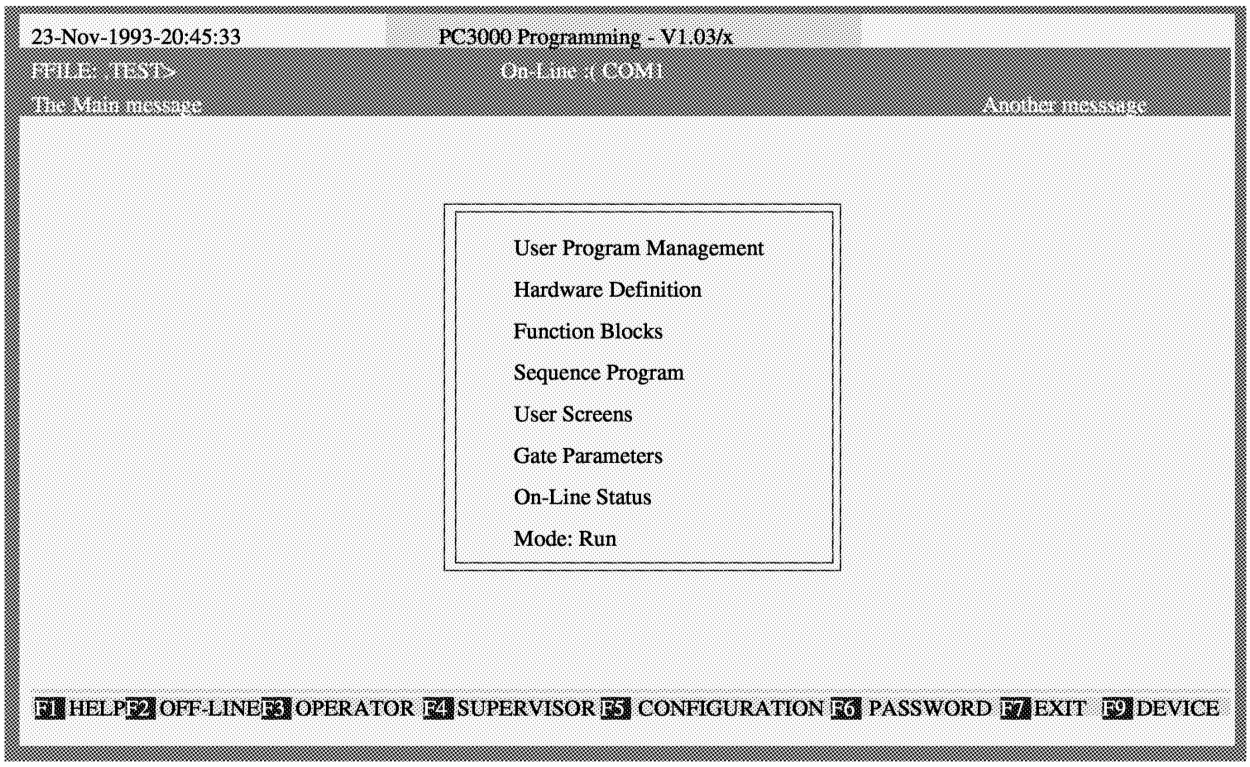
The parameter **S_Message** holds the secondary message to be displayed on the right half of the display line. The input string can be up to 40 characters long.

Example:

Messages **P_Message**:= 'The main message';

Messages **S_Message:**= 'Another message';

Results in a Programming Station screen (DOS based PS)



Parameter Attributes

Name	Type	Cold Start	Read Access	Write Access	Type Specific Information	
P_Message	**STRING**		Oper	Config	Max String Length	40 characters
S_Message	**STRING**		Oper	Config	Max String Length	40 Characters

Table 2-2 Messages Parameter Attributes

TASK FUNCTION BLOCK

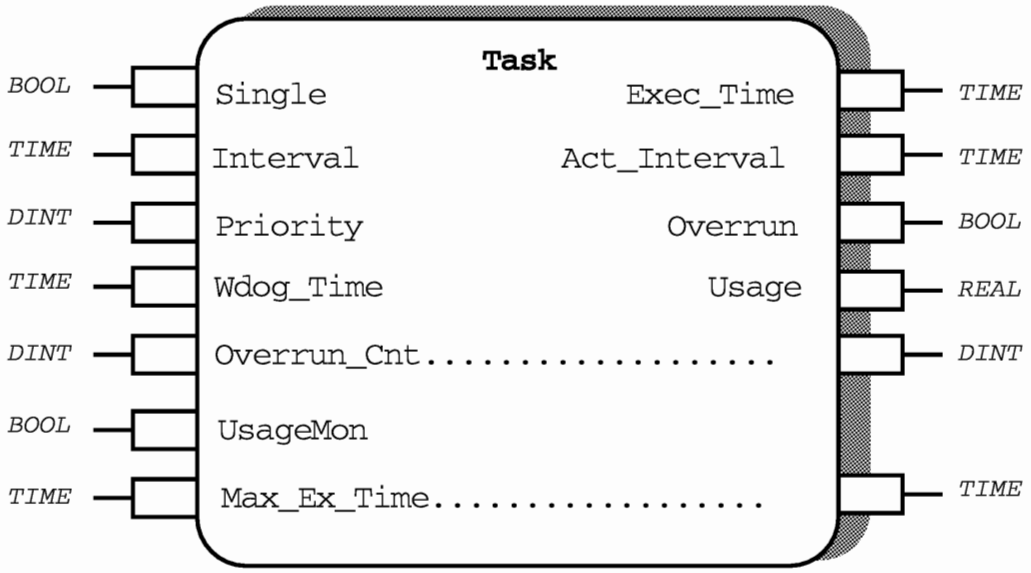


Figure 2-3 Task Function Block Diagram

Functional Description

The Task function block provides the user interface to the scheduler within the PC3000 system. Refer to the chapter Real Time Task Scheduler, in the PC3000 Real Time Operating System Reference. All user programs have two default tasks blocks with 10ms and 100ms intervals.

Function Block Attributes

Type: ..... C 10

Class: ..... SYSTEM

Default_Task: ..... Task_2

Short List: ..... Interval, Act_Interval, Overrun_Cnt,

Instance Memory Requirement: ..... 4396 Bytes

Parameter Descriptions

Single (SGL)

The parameter **Single** has no function It has been included for future enhancement. It should be left at its default value of No (0)

Interval (IV)

The **Interval** parameter defines the desired sample time between scheduling of the task. The fastest task must have an Interval of no greater than 65 ms. In addition,

Peach task (except the fastest task) must have an Interval which is an integer multiple of the next faster task, i.e. task Intervals must be rate monotonic. If the fastest task is slower than 65 ms or the intervals are not rate monotonic, then the PC3000 will be unable to run and a system error will be registered when the RUN command is issued. The PC3000 will remain in the HALTED state.

Priority (PRI)

The parameter **Priority** defines the priority of the task for the pre-emptive scheduler to use, where 0 is the highest priority and 7 is the lowest. The fastest task must have the highest priority.

Wdog_Time (WDT)

Wdog_Time defines the maximum amount of time that the function block can take to execute before a watch-dog is registered. If the task has not completed running **Wdog_Time** after its scheduled start time, then a user watchdog will be registered, causing a system error to be logged and the PC3000 to reset.

Overrun_Cnt (OVC)

The parameter **Overrun_Cnt** indicates the number of overruns of the task that have occured since the last time **Overrun_Cnt** was cleared. **Overrun_Cnt** is cleared by assigning it the value 0 or by restarting the user program.

Exec_Time (EXT)

Exec_Time indicates the amount of time taken by the task for its last execution. This includes the time to execute routines which interrupted execution of the task, but does not include the execution time of other tasks which pre-empted this one.

Act_Interval (AIV)

Act_Interval indicates the last attained interval between executions. When the system is not overrunning this will be equal to **Interval.** If an overrun should occur, **Act_Interval** will indicate the actual interval achieved.

Overrun (VRN)

Overrun indicates whether the task execution time is greater than Interval. In normal operation **Act_Interval** is less than **Interval,** so **Overrun** will be set to No (0). If **Act_Interval** should be greater than **Interval Overrun** will be set to Yes (1).

UsageMon (USM) [Version 3.0 onwards]

The parameter **UsageMon** is used to enable and disable the calculations for the % usage of the task function block i.e. the % of the available processing time for this task which is currently being used. When **UsageMon** is set to On the calculation is performed; when **UsageMon** is set to Off the calculation is not performed leaving more processing time available for other functions.

Usage (USG) [Version 3.0 onwards]

When **UsageMon** is set to On, **Usage** indicates the % usage of the task i.e. the % of the available processing time for this task which is currently being used.

Max_Ex_Time (MEX) [Version 3.0 onwards]

Max_Ex_Time indicates the maximum execution time for this task. It holds the highest value recorded for the **Exec_Time** parameter since **Max_Ex_Time** was last reset by the user. **Max_Ex_Time** is an Input/Output parameter which is updated by the block. It can also be reset by the user program or from the programming station in the On-line mode. This allows the peak loading of tasks to be simply monitored for debugging and performance optimisation.

Parameter Attributes

Name	Type	Cold Start	Read Access	Write Access	Type Specific Information	
Act_Interval	**TIME**	0ms	Oper			
Exec_Time	**TIME**	0ms	Oper			
Interval	**TIME**	500ms	Oper	Super	High Limit Low Limit	5m 5ms
Overrun	**BOOL**	No (0)	Oper		Senses	No (0) Yes (1)
Overrun_Cnt	**DINT**	0	Oper	Super		
Priority	**DINT**		Oper	Super	High Limit Low Limit	7 0
Single	**BOOL**	No (0)	Oper	Super	Senses:	No (0) Yes (1)
Wdog_Time	**TIME**	5s	Oper	Super	High Limit Low Limit	10m 0ms
UsageMon	**BOOL**	Off (0)	Oper	Oper	Senses	Off (0) On (1)
Usage	**REAL**	0	Oper	Block	High Limit Low Limit	 0
Max_Ex_Time	**TIME**	0ms	Oper	Oper	High Limit Low Limit	 0ms

Table 2-3 Tasks Parameter Attributes

RT_CLOCK FUNCTION BLOCK

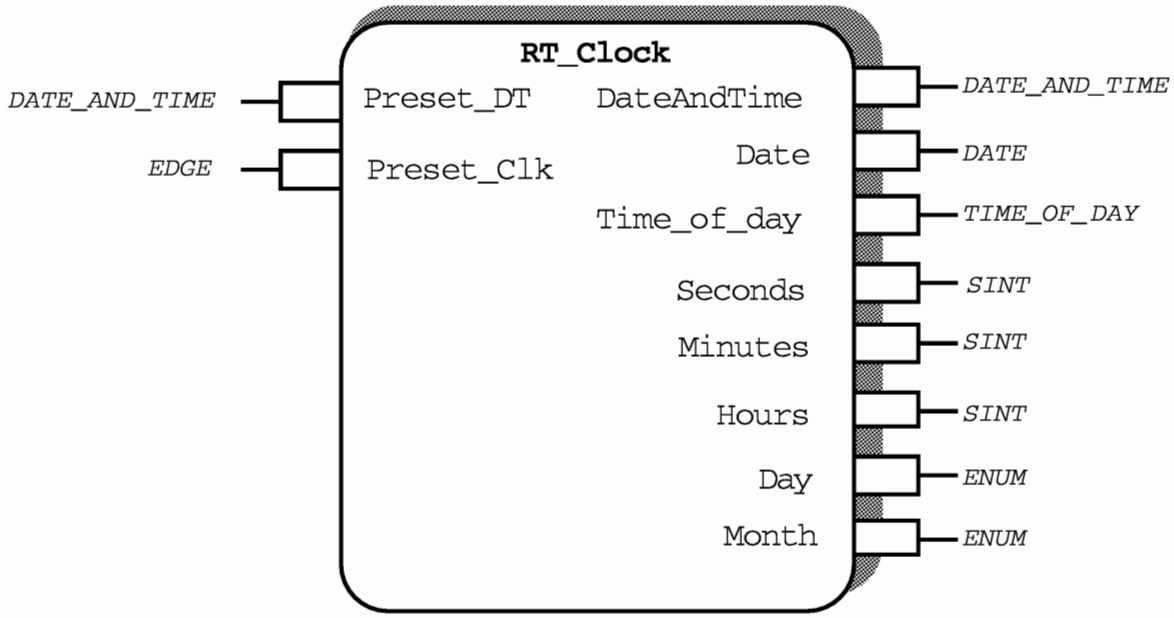


Figure 2-4 RT_Clock Function Block Diagram

Functional Description

The **RT_Clock** function block allows reading and setting of the PC3000's internal real time clock. To set the clock, the correct date and time should be input to **Preset_DT.** This is then entered into the PC3000 by changing **Preset_Clk** from Tock (0) to Tick (1), with the resetting of the clock taking place on the rising edge of the change, i.e. on the transition from 0 to 1. From version 3.0 onwards, **Preset_Clk** automatically returns to Tock (0) after the clock has been set. In earlier versions **Preset_Clk** must be manually reset to Tock (0) on completion.

Function Block Attributes

Type:..... 8 48

Class: .....SYSTEM

Default Task: .....Task_2

Short_List: .....DateAndTime

Instance Memory Reqirement: .....40 Bytes

Parameter Descriptions

Preset_DT (PDT)

The parameter **Preset_DT** is used only for presetting the date and time into the PC3000 clock. The value of **Preset_DT** is entered on the rising edge of changing **Preset_Clk** from Tock (0) to Tick (1)

Preset_Clk (PC)

The parameter **Preset_Clk** is used to enter **Preset_DT** into the PC3000. The entry takes place on the rising edge of changing **Preset_Clk** from Tock (0) to Tick (1).

DateAndTime (DAT)

Date And Time is the current date and time. This is represented in seconds from 1st January 1970, so dates before 1st January 1970 are invalid.

Date (D)

Date is the current date, as read from the memory of the PC3000.

Time_Of_Day (TOD)

Time_Of_Day is the current time, represented as a 24 hour clock measurement.

Seconds (SEC)

The parameter **Seconds** outputs the current time in seconds, from 0 to 59.

Minutes (MIN)

The parameter **Minutes** outputs the current time in minutes, from 0 to 59.

Hours (HR)

The parameter **Hours** outputs the current time in hours, from 0 to 23.

Day (DAY)

The parameter **Day** outputs the current date in days, from 0 to 6 (Sunday to Saturday).

Month (M)

The parameter **Month** outputs the current date in months, from 0 to 11 (January to December).

Parameter Attributes

Name	Type	Cold Start	Read Access	Write Access	Type Specific Information	
Date	DATE	Jan 1 1970	Oper	Block	High Limit Low Limit	Jan 19 2038 Jan 1 1970
DateAndTime	DATE_ AND_ TIME	Jan 1 1970 00:00:00	Oper	Block	High Limit Low Limit	Jan 19 2038 03:14:06 Jan 1 1970 00:00:00
Day	ENUM	Sun (0)	Oper	Block	Senses	Sun (0) Mon (1) Tues (2) Wed (3) Thurs (4) Fri (5) Sat (6)
Hours	SINT	0	Oper	Block	High Limit Low Limit	23 0
Minutes	SINT	0	Oper	Block	High Limit Low Limit	59 0
Month	ENUM	Jan (0)	Oper	Block	Enumerated Values	Jan (0) Feb (1) March (2) April (3) May (4) June (5) July (6) Aug (7) Sept (8) Oct (9) Nov (10) Dec (11)
Preset_Clk	BOOL	Tock (0)	Oper	Super	Senses	Tock (0) Tick (1)
Preset_DT	DATE_ AND_ TIME	Jan 1 1970 00:00:00	Super	Super	High Limit Low Limit	Jan 19 2038 03:14:06 Jan 1 1970 00:00:00
Seconds	SINT	0	Oper	Block	High Limit Low Limit	59 0
Time_Of_Day	TIME_ OF_DAY	00:00:00	Oper	Block	High Limit Low Limit	23:59:59 00:00:00

Table 2-4 RT Clock Parameter Attributes