Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
Chipsmall
采购部

TERIDIAN SEMICONDUCTOR CORP.

A Maxim Integrated Products Brand

71M6511/71M6511H
Single-Phase Energy Meter IC

DATA SHEET

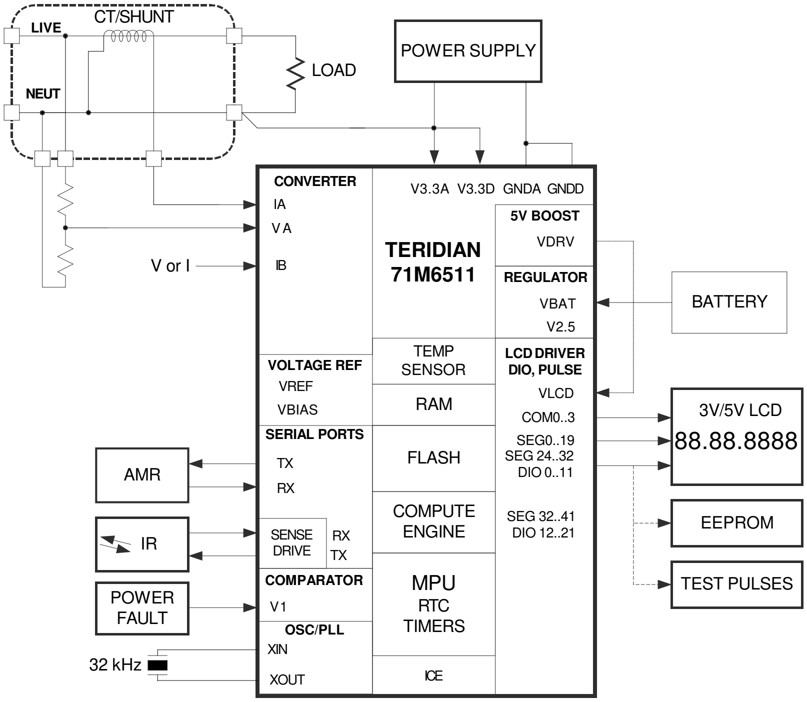
NOVEMBER 2010

GENERAL DESCRIPTION

The 71M6511 is a highly integrated SOC with an MPU core, RTC, flash, and LCD driver. Our Single Converter Technology® with a 22-bit delta-sigma ADC, three analog inputs, digital temperature compensation, precision voltage reference, and 32-bit computation engine (CE) supports a wide range of single-phase metering applications with very few low cost external components. A 32kHz crystal time base for the entire system and internal battery backup support for RAM and RTC further reduce system cost.

Maximum design flexibility is supported with multiple UARTs, I^2C, a power fail comparator, a 5V LCD charge pump, up to 12 DIO pins and an in-system programmable flash. The device is offered in high (0.1%) and standard (0.5%) accuracy versions for multifunction residential/commercial meter applications requiring multiple voltage/current inputs and complex LCD or DIO configurations.

A complete array of ICE and development tools, programming libraries and reference designs enable rapid development and certification of meters that meet most demanding worldwide electricity metering standards.



7/20/2007

Single Converter Technology is a registered trademark of Maxim Integrated Products, Inc.

FEATURES

- Wh accuracy < 0.1% over 2000:1 range
- Exceeds IEC 62053/ANSIC 12.20
- Voltage reference
  < 10ppm/°C -- 71M6511H,
  < 50ppm/°C -- 71M6511
- Three sensor inputs - V_{DD} referenced
- Low jitter Wh/VARh pulse outputs
- Pulse count for pulse outputs
- Four-quadrant metering
- Voltage/current angle
- Line frequency count for RTC
- Digital temperature compensation
- Sag detection
- Independent 32-bit compute engine
- 40-70Hz line frequency range with same calibration
- Phase compensation (±7°)
- Battery backup for RAM and RTC
- 22mW at 3.3V, 7.2μW backup
- Flash memory option with security
- 22-bit delta-sigma ADC
- 8-bit MPU (80515) - 1 clock cycle per instruction
- LCD driver (≤ 128 pixels)
- High speed SSI serial output
- RTC for time-of-use functions
- Hardware watchdog timer
- Up to 12 general-purpose I/O pins
- 64KB flash, 7KB RAM
- Two UARTs for IR and AMR
- 64-lead LQFP package

Table of Contents

Figures

Tables

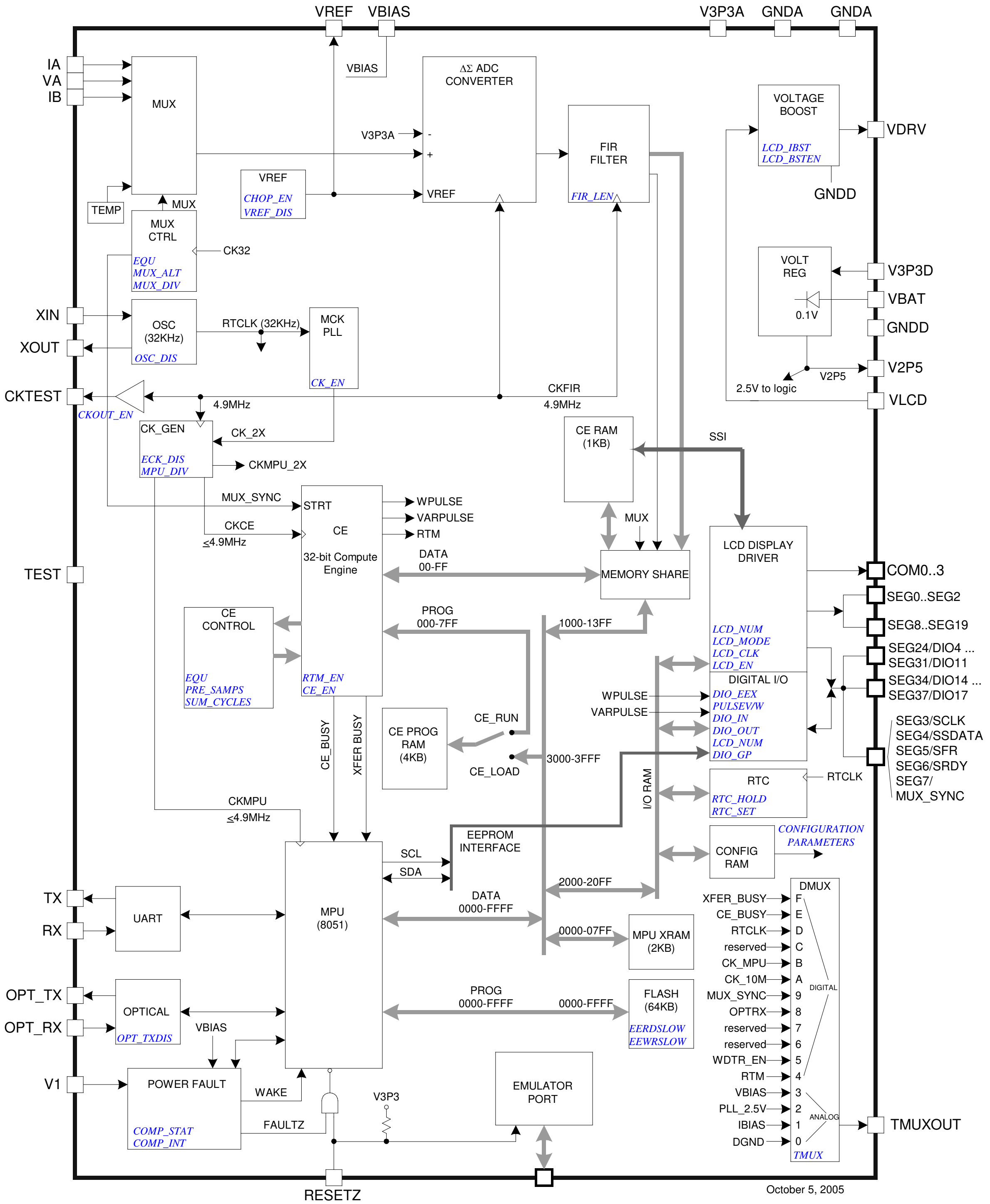


Figure 1: IC Functional Block Diagram

HARDWARE DESCRIPTION

Hardware Overview

The TERIDIAN 71M6511 single chip single-phase meter integrates all primary functional blocks required to implement a solid-state electricity meter. Included on chip are an analog front end (AFE), an 8051-compatible microprocessor (MPU) which executes one instruction per clock cycle (80515), an independent 32-bit digital computation engine (CE), a voltage reference, a temperature sensor, LCD drivers, RAM, flash memory, a real time clock (RTC), and a variety of I/O pins. Various current sensor technologies are supported including Current Transformers (CT), Resistive Shunts, and Rogowski (*di/dt)* Coils.

In addition to advanced measurement functions, the real time clock function allows the 71M6511/6511H to record time of use (TOU) metering information for multi-rate applications. Measurements can be displayed on either a 3V or a 5V LCD. Flexible mapping of LCD display segments will facilitate integration with any LCD format. The design trade-off between the number of LCD segments and DIO pins can be flexibly configured using memory-mapped I/O to accommodate various requirements.

The 71M6511 includes several I/O peripheral functions that improve the functionality of the device and reduce the component count for most meter applications. The I/O peripherals include two UARTs, digital I/O, comparator inputs, LCD display drivers, I^{2}C interface and an optical/IR interface.

One of the two internal UARTs (UART1) is adapted to support an Infrared LED with internal drive output and sense input but it can also function as a standard UART.

A block diagram of the chip is shown in Figure 1. A detailed description of various hardware blocks follows.

Analog Front End (AFE)

The AFE of the TERIDIAN 71M6511 Power Meter IC is comprised of an input multiplexer, a delta-sigma A/D converter with a voltage reference, followed by an FIR filter. A block diagram of the AFE is shown in Figure 3.

Multiplexer

The input multiplexer supports four input signals that are applied to the pins IA, VA, and IB plus the output of the internal temperature sensor. The multiplexer can be operated in two modes:

- During a normal multiplexer cycle, the signals from the pins IA, VA, and IB, are selected.
- During the alternate multiplexer cycle, the temperature signal (TEMP) is selected, along with the other signal sources shown in Table 1.

Alternate multiplexer cycles are usually performed infrequently (every second or so). VA is not replaced in the alternate multiplexer cycles. Missing samples due to alternate multiplexer cycles are automatically interpolated by the CE.

EQU	**Channels used from MUX Sequence** States 0 → 3				**Channels used from alternative MUX Sequence** States 0 → 3			
	0	1	2	3	0	1	2	3
000	IA	VA	IB	-	TEMP	VA	-	-
001	IA	VA	IB	-	TEMP	VA	IB	-

Table 1: Inputs Selected in Regular and Alternate Multiplexer Cycles

In a typical application, the IA input is connected to a current transformer that senses the line current. VA is typically connected to a voltage sensor through resistor dividers. IB may be connected to a second current transformer, e.g. for optional tamper detection.

The Multiplexer Control Circuit handles the setting of the multiplexer. The function of the Multiplexer Control Circuit is governed by the I/O RAM registers *MUX_ALT* (0x2005[2]), *EQU* (0x2000[7:5]), and *MUX_DIV* (0x2002[7:6]). *MUX_DIV* controls the number of samples per cycle. It can request 2, 3, 4, or 6 multiplexer states per cycle.

The *MUX_ALT* bit requests an alternate multiplexer cycle. The bit may be asserted on any MPU cycle and may be subsequently de-asserted on any cycle including the next one. A rising edge on *MUX_ALT* will cause the Control Circuit to wait until the next multiplexer cycle and implement a single alternate cycle.

Multiplexer Control Circuit also controls the FIR filter initiation and the chopping of the ADC reference voltage, VREF. The Multiplexer Control Circuit is clocked by CK32, the 32768Hz clock from the PLL block, and launches each pass through the CE program.

Table 2 shows the possible settings for *MUX_DIV* and *FIR_LEN* and the resulting channels sampled along with sample frequencies.

MUX_DIV **(0x2002[7.6])**	**Number of channels selected (mux states per cycle)**	**Number of CK32 states for code pass**	**Effective sample frequency [Hz]**	**Number of CK32 states for code pass**	**Effective sample frequency [Hz]**
		FIR_LEN **= 0**		*FIR_LEN* **=1**	
00	---	Not Allowed			
01	**4**	9	3640.89	**13**	**2520.615**
10	3	7	4681.143	10	3276.8
11	2	5	6553.6	7	4681.143

Table 2: Channel control based on *MUX_DIV* and *FIR_LEN*

ADC

A single 21/22-bit delta-sigma A/D converter (ADC) digitizes the power inputs to the AFE. The resolution of the ADC is programmable using the I/O RAM register *FIR_LEN* register (0x2005[4]). ADC resolution may be selected to be 21 bits (*FIR_LEN*=0), or 22 bits (*FIR_LEN*=1). Conversion time is two cycles of CK32 with *FIR_LEN* = 0 and three cycles with *FIR_LEN* = 1.

In order to provide the maximum resolution, the ADC should be operated with *FIR_LEN* = 1. Accuracy, timing and functional specifications in this data sheet are based on *FIR_LEN* = 1 and *MUX_DIV* = 1 (four CK32 cycles). Alternative specifications are also provided for *FIR_LEN* = 1 and *MUX_DIV* = 2 (three CK32 cycles) in the CE Program and Environment section.

Initiation of each ADC conversion is controlled by the Multiplexer Control Circuit as described previously.

FIR Filter

The finite impulse response (FIR) filter is an integral part of the ADC and it is optimized for use with the multiplexer. The purpose of the FIR is to decimate the ADC output to the desired resolution. At the end of each ADC conversion, the output data of the FIR filter (raw data) is stored into the CE DRAM location determined by the multiplexer selection. The location of the raw data in the CE DRAM is specified in the CE Program and Environment Section.

Voltage Reference

The 71M6511/6511H includes an on-chip precision bandgap voltage reference that incorporates auto-zero techniques. The reference of the 71M6511H is trimmed in production to minimize errors caused by component mismatch and drift. The result is a voltage output with a predictable temperature coefficient.

The voltage reference is chopper stabilized, i.e. the polarity can be switched by the MPU using the I/O RAM register *CHOP_ENA* (0x2002[5:4]). The two bits in the *CHOP_ENA* register enable the MPU to operate the chopper circuit in regular or inverted operation, or in “toggling” mode. When the chopper circuit is toggled in between multiplexer cycles, DC offsets on the measured signals will automatically be averaged out.

The general topology of a chopped amplifier is given in Figure 2.

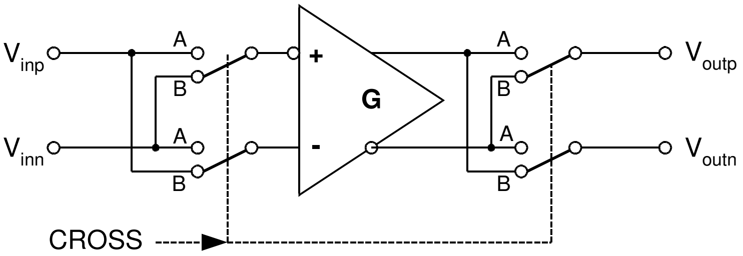


Figure 2: General Topology of a Chopped Amplifier

It is assumed that an offset voltage Voff appears at the positive amplifier input. With all switches, as controlled by CROSS in the “A” position, the output voltage is:

Voutp – Voutn = G (Vinp + Voff – Vinn) = G (Vinp – Vinn) + G Voff

With all switches set to the “B” position by applying the inverted CROSS signal, the output voltage is:

Voutn – Voutp = G (Vinn – Vinp + Voff) = G (Vinn – Vinp) + G Voff, or

Voutp – Voutn = G (Vinp – Vinn) - G Voff

Thus, when CROSS is toggled, e.g. after each multiplexer cycle, the offset will alternately appear on the output as positive and negative, which results in the offset effectively being eliminated, regardless of its polarity or magnitude.

The Functional Description Section contains a chapter with a detailed description on controlling the *CHOP_ENA* register.

Temperature Sensor

The 71M6511/6511H includes an on-chip temperature sensor implemented as a bandgap reference. It is used to determine the die temperature The MPU may request an alternate multiplexer cycle containing the temperature sensor output by asserting *MUX_ALT*.

The primary use of the temperature data is to determine the magnitude of compensation required to offset the thermal drift in the system (see section titled “Temperature Compensation”).

The zero reference for the temperature sensor is VBIAS.

Functional Description

The AFE functions as a data acquisition system, controlled by the MPU. The main signals (IA, VA, IB) are sampled and the ADC counts obtained are stored in CE RAM where they can be accessed by the CE and, if necessary, by the MPU. Alternate multiplexer cycles are initiated less frequently by the MPU to gather access to the slow temperature signal.

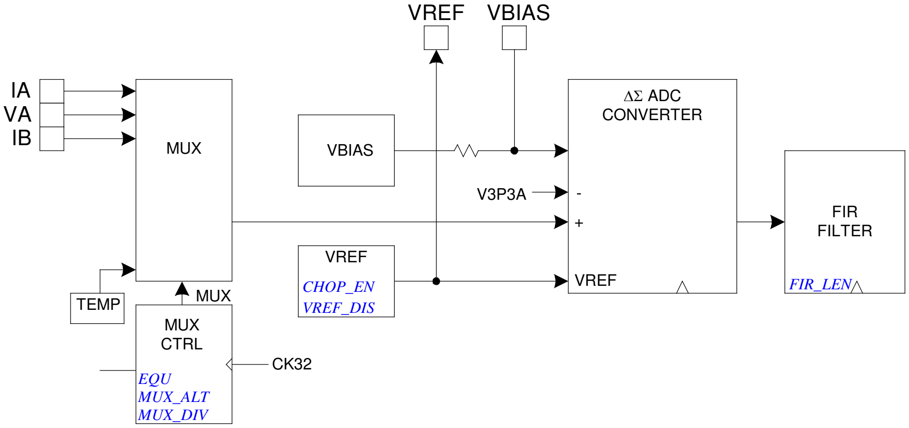


Figure 3: AFE Block Diagram

Computation Engine (CE)

The CE, a dedicated 32-bit RISC processor, performs the precision computations necessary to accurately measure energy. The CE calculations and processes include:

- Multiplication of each current sample with its associated voltage sample to obtain the energy per sample (when multiplied with the constant sample time).
- Frequency-insensitive delay cancellation on all six channels (to compensate for the delay between samples caused by the multiplexing scheme).
- 90° phase shifter (for VAR calculations).
- Pulse generation.
- Monitoring of the input signal frequency (for frequency and phase information).
- Monitoring of the input signal amplitude (for sag detection).
- Scaling of the processed samples based on chip temperature (temperature compensation) and calibration coefficients.

The CE program RAM (CE PRAM) is loaded at boot time by the MPU and then executed by the CE. Each CE instruction word is 2 bytes long. The CE program counter begins a pass through the CE code each time multiplexer state 0 begins. The code pass ends when a HALT instruction is executed. For proper operation, the code pass must be completed before the multiplexer cycle ends (see System Timing Summary in the Functional Description Section).

The CE data RAM (CE DRAM) can be accessed by the FIR filter block, the RTM circuit, the CE, and the MPU. Assigned time slots are reserved for FIR, RTM, and MPU, respectively, such that memory accesses to CE_RAM do not collide. Holding registers are used to convert 8-bit wide MPU data to/from 32-bit wide CE DRAM data, and wait states are inserted as needed, depending on the frequency of CKMPU.

Table 3 shows the CE DRAM addresses allocated to analog inputs from the AFE.

Address (hex)	Name	Description
0x00	IA	Phase A current
0x01	VA	Phase A voltage
0x02	IB	Phase B current
0x03	-	Reserved
0x04	-	Reserved
0x05	-	Reserved
0x06	TEMP	Temperature
0x07	--	Reserved

Table 3: CE DRAM Locations for ADC Results

Meter Equations

The Compute Engine (CE) program for residential meter configurations implements the equations in Table 4. The I/O RAM register *EQU* specifies the equation to be used based on the number and arrangement of phases used for metering. In case of single-phase metering, the unconnected input should be tied to V3P3A, the analog supply voltage. The *EQU* selection enables the 71M6511 to calculate single-phase power measurement based on the type of service used. Table 4 also states the sequence of the multiplexer in the AFE.

EQU	**Formula**	**Channels used from MUX Sequence** States 0 → 3				**Channels used from alternative MUX Sequence** States 0 → 3			
		0	1	2	3	0	1	2	3
000	**VA IA** **(1 element, 2W 1ϕ)**	IA	VA	IB	-	TEMP	VA	-	-
001	**VA(IA-IB)/2** **(1 element, 3W 1ϕ)**	IA	VA	IB	-	TEMP	VA	IB	-

Table 4: Standard Meter Equations (inputs shown gray are scanned but not used for calculation)

Pulse Generator

The CE contains two pulse generators which create low jitter pulses at a rate set by the CE DRAM registers *APULSEW*WRATE* and *APULSER*WRATE* if *EXT_PULSE* (a CE input variable in CE DRAM) is 15. This mode puts the MPU in control of pulse generation by placing values into the *APULSEW* and *APULSER* registers ("external pulse generation").

If *EXT_PULSE* is 0, *APULSEW* is replaced with *WSUM_X* and *APULSER* is replaced with *VARSUM_X*. In this mode, the CE generates pulse based on its internal computation of *WSUM_X* and *VARSUM_X*, the signed sums of energy from all three elements ("internal pulse generation").

The *DIO_PV* and *DIO_PW* bits as described in the Digital I/O section can be programmed to route WPULSE and VARPULSE to the output pins DIO6 and DIO7 respectively. DIO6 and DIO7 can be configured to generate interrupts (useful for pulse counting by the MPU – see On-Chip Resources (DIO Section).

Real-Time Monitor

The CE contains a Real Time Monitor (RTM), which can be programmed to monitor four selectable CE RAM locations at full sample rate. The four monitored locations are serially output to the TMUXOUT pin via the digital output multiplexer at the beginning of each CE code pass (see the Test Ports Section for details)

CE Functional Overview

The ADC processes one sample per channel per multiplexer cycle. Figure 4 shows the timing of the samples taken during one multiplexer cycle.

The number of samples processed during one accumulation cycle is controlled by the I/O RAM registers *PRE_SAMPS* (0x2001[7:6]) and *SUM_CYCLES* (0x2001[5:0]). The integration time for each energy output is

PRE_SAMPS * *SUM_CYCLES* / 2520.6, where 2520.6 is the sample rate [Hz] (for *MUX_DIV* = 1)

For example, *PRE_SAMPS* = 42 and *SUM_CYCLES* = 50 will establish 2100 samples per accumulation cycle. *PRE_SAMPS* = 100 and *SUM_CYCLES* = 21 will result in the exact same accumulation cycle of 2100 samples or 833ms. After an accumulation cycle is completed, the XFER_BUSY interrupt signals to the MPU that accumulated data are available.

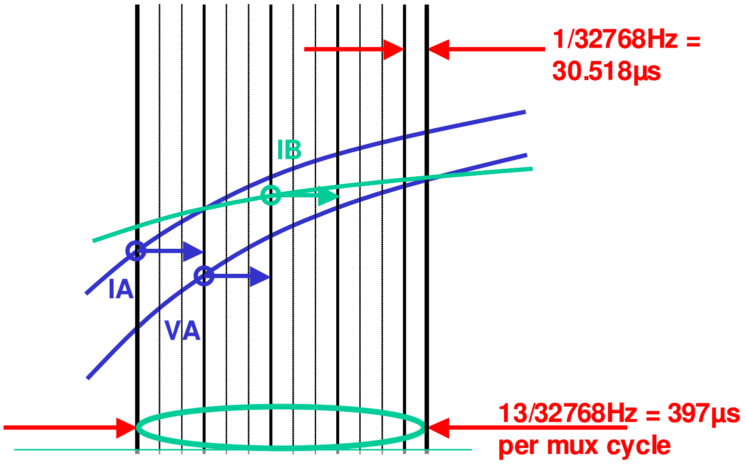


Figure 4: Samples in Multiplexer Cycle

The end of each multiplexer cycle is signaled to the MPU by the CE_BUSY interrupt. At the end of each multiplexer cycle, status information, such as sag data and the digitized input signal, is available to the MPU.

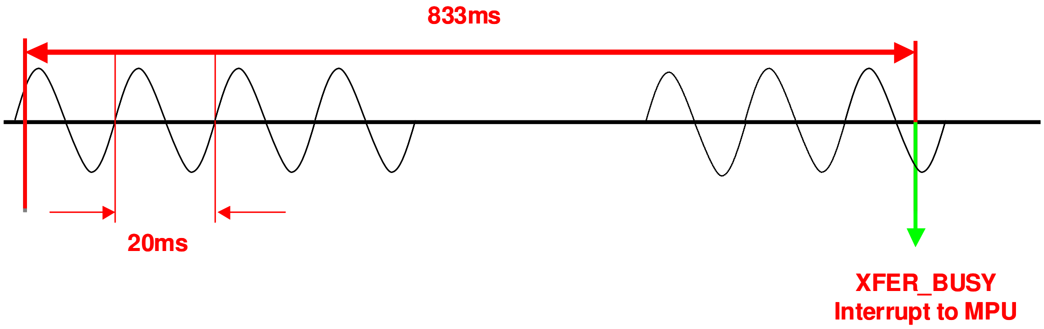


Figure 5: Accumulation Interval

Figure 5 shows the accumulation interval resulting from *MUX_DIV* = 1, *PRE_SAMPS* = 42 and *SUM_CYCLES* = 50, consisting of 2100 samples of 397µs each, followed by the XFER_BUSY interrupt. The sampling in this example is applied to a 50Hz signal.

There is no correlation between the line signal frequency and the choice of *PRE_SAMPS* or *SUM_CYCLES (*even though when *SUM_CYCLES* = 42 one set of *SUM_CYCLES* happens to sample a period of 16.6ms). Furthermore, sampling does not have to start when the line voltage crosses the zero line.

Delay Compensation

When measuring the energy of a phase (i.e., Wh and VARh) in a service, the voltage and current for that phase must be sampled at the same instant. Otherwise, the phase difference, Φ, introduces errors.

$$\phi = \frac{t_{delay}}{T} \cdot 360^{o} = t_{delay} \cdot f \cdot 360^{o}$$

Where *f* is the frequency of the input signal and t_{delay} is the sampling delay between voltage and current.

In traditional meter ICs, sampling is accomplished by using two A/D converters per phase (one for voltage and the other one for current) controlled to sample simultaneously. Teridian's Single-Converter Technology®, however, exploits the 32-bit signal processing capability of its CE to implement "constant delay" all-pass filters. These all-pass filters correct for the conversion time difference between the voltage and the corresponding current samples that are obtained with a single multiplexed A/D converter.

The "constant delay" all-pass filters provide a broad-band delay β that is precisely matched to the difference in sample time between the voltage and the current of a given phase. This digital filter does not affect the amplitude of the signal, but provides a precisely controlled phase response. The delay compensation implemented in the CE aligns the voltage samples with their corresponding current samples by routing the voltage samples through the all-pass filter, thus delaying the voltage samples by β, resulting in the residual phase error β – Φ. The residual phase error is negligible, and is typically less than ±1.5 milli-degrees at 100Hz, thus it does not contribute to errors in the energy measurements.

80515 MPU Core

80515 Overview

The 71M6511/6511H includes an 80515 MPU (8-bit, 8051-compatible) that processes most instructions in one clock cycle. Using a 5MHz clock results in a processing throughput of 5 MIPS. The 80515 architecture eliminates redundant bus states and implements parallel execution of fetch and execution phases. Normally a machine cycle is aligned with a memory fetch, therefore, most of the 1-byte instructions are performed in a single cycle. This leads to an 8x performance (in average) improvement (in terms of MIPS) over the Intel 8051 device running at the same clock frequency.

Actual processor clocking speed can be adjusted to the total processing demand of the application (metering calculations, AMR management, memory management, LCD driver management and I/O management) using the I/O RAM register *MPU_DIV[2:0]*.

Typical measurement and metering functions based on the results provided by the internal 32-bit compute engine (CE) are available for the MPU as part of TERIDIAN's standard library. A standard ANSI "C" 80515-application programming interface library is available to help reduce design cycle.

Memory Organization

The 80515 MPU core incorporates the Harvard architecture with separate code and data spaces.

Memory organization in the 80515 is similar to that of the industry standard 8051. There are three memory areas: Program memory (flash), external data memory (XRAM), physically consisting of XRAM, CE DRAM, CE PRAM and I/O RAM, and internal data memory (Internal RAM). Figure 6 shows the memory map (see also Table 55).

Internal and External Data Memory: Both internal and external data memory are physically located on the 71M6511 IC. External data memory is only external to the 80515 MPU core.

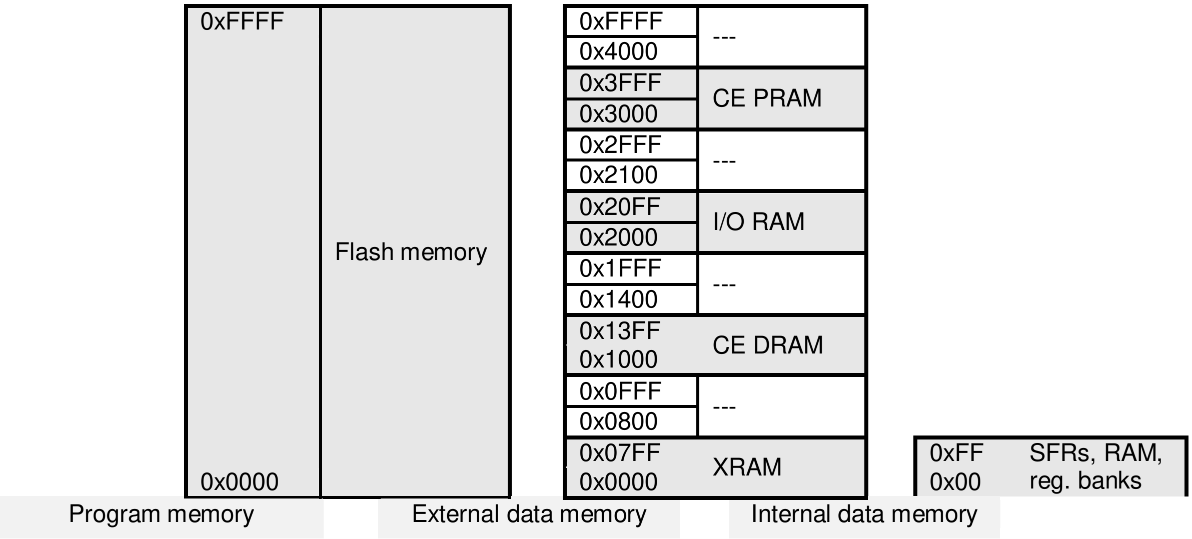


Figure 6: Memory Map

Program Memory: The 80515 can address up to 64KB of program memory space from 0x0000 to 0xFFFF. Program memory is read when the MPU fetches instructions or performs a MOVC operation.

After reset, the MPU starts program execution from location 0x0000. The lower part of the program memory includes reset and interrupt vectors. The interrupt vectors are spaced at 8-byte intervals, starting from 0x0003.

External Data Memory: While the 80515 can address up to 64KB of external data memory in the space from 0x0000 to 0xFFFF, only the memory ranges shown in Figure 6 contain physical memory. The 80515 writes into external data memory when the MPU executes a MOVX @Ri,A or MOVX @DPTR,A instruction. The MPU reads external data memory by executing a MOVX A,@Ri or MOVX A,@DPTR instruction (SFR USR2 provides the upper 8 bytes for the MOVX A,@Ri instruction).

Clock Stretching: MOVX instructions can access fast or slow external RAM and external peripherals. The three low ordered bits of the CKCON register define the stretch memory cycles. Setting all the *CKCON* stretch bits to one allows access to very slow external RAM or external peripherals.

Table 5 shows how the signals of the External Memory Interface change when stretch values are set from 0 to 7. The widths of the signals are counted in MPU clock cycles. The post-reset state of the CKCON register, which is in bold in the table, performs the MOVX instructions with a stretch value equal to 1.

CKCON register			**Stretch Value**	**Read signals width**		**Write signal width**	
CKCON.2	CKCON.1	CKCON.0		memaddr	memrd	memaddr	memwr
0	0	0	0	1	1	2	1
0	**0**	**1**	**1**	**2**	**2**	**3**	**1**
0	1	0	2	3	3	4	2
0	1	1	3	4	4	5	3
1	0	0	4	5	5	6	4
1	0	1	5	6	6	7	5
1	1	0	6	7	7	8	6
1	1	1	7	8	8	9	7

Table 5: Stretch Memory Cycle Width

There are two types of instructions, differing in whether they provide an eight-bit or sixteen-bit indirect address to the external data RAM.

In the first type (MOVX A,@Ri), the contents of R0 or R1, in the current register bank, provide the eight lower-ordered bits of address. The eight high-ordered bits of address are specified with the USR2 SFR. This method allows the user paged access (256 pages of 256 bytes each) to the full 64KB of external data RAM. In the second type of MOVX instruction (MOVX A,@DPTR), the data pointer generates a sixteen-bit address. This form is faster and more efficient when accessing very large data arrays (up to 64 Kbytes), since no additional instructions are needed to set up the eight high ordered bits of address.

It is possible to mix the two MOVX types. This provides the user with four separate data pointers, two with direct access and two with paged access to the entire 64KB of external memory range.

Dual Data Pointer: The Dual Data Pointer accelerates the block moves of data. The standard DPTR is a 16-bit register that is used to address external memory or peripherals. In the 80515 core, the standard data pointer is called DPTR, the second data pointer is called DPTR1. The data pointer select bit chooses the active pointer. The data pointer select bit is located at the LSB of the DPS register (DPS.0). DPTR is selected when DPS.0 = 0 and DPTR1 is selected when DPS.0 = 1.

The user switches between pointers by toggling the LSB of the DPS register. All DPTR-related instructions use the currently selected DPTR for any activity.

The second data pointer may not be supported by certain compilers.

Internal Data Memory: The Internal data memory provides 256 bytes (0x00 to 0xFF) of data memory. The internal data memory address is always 1 byte wide and can be accessed by either direct or indirect addressing. The Special Function Registers occupy the upper 128 bytes**. This SFR area is available only by direct addressing. Indirect addressing accesses the upper 128 bytes of Internal RAM.**

The lower 128 bytes contain working registers and bit-addressable memory. The lower 32 bytes form four banks of eight registers (R0-R7). Two bits on the program memory status word (PSW) select which bank is in use. The next 16 bytes form a block of bit-addressable memory space at bit addressees 0x00-0x7F. All of the bytes in the lower 128 bytes are accessible through direct or indirect addressing. Table 6 shows the internal data memory map.

Address	Direct addressing	Indirect addressing
0xFF	Special Function Registers (SFRs)	RAM
0x80		
0x7F	Byte-addressable area	
0x30		
0x2F	Bit-addressable area	
0x20		
0x1F	Register banks R0…R7	
0x00		

Table 6: Internal Data Memory Map

Special Function Registers (SFRs)

A map of the Special Function Registers is shown in Table 7.

Hex\Bin	Bit-addressable	Byte-addressable							Bin/Hex
	X000	X001	X010	X011	X100	X101	X110	X111	
F8	**INTBITS**								FF
F0	B								F7
E8	**WDI**								EF
E0	A								E7
D8	WDCON								DF
D0	PSW								D7
C8									CF
C0	IRCON								C7
B8	IEN1	IP1	S0RELH	S1RELH				USR2	BF
B0			**FLSHCTL**					**PGADR**	B7
A8	IEN0	IP0	S0RELL						AF
A0	**P2**	**DIR2**	**DIR0**						A7
98	S0CON	S0BUF	IEN2	S1CON	S1BUF	S1RELL	**EEDATA**	**EECTRL**	9F
90	**P1**	**DIR1**	DPS		**ERASE**				97
88	TCON	TMOD	TL0	TL1	TH0	TH1	CKCON		8F
80	**P0**	SP	DPL	DPH	DPL1	DPH1	WDTREL	PCON	87

Table 7: Special Function Registers Locations

Only a few addresses are occupied, the others are not implemented. SFRs specific to the 651X are shown in **bold** print. Any read access to unimplemented addresses will return undefined data, while any write access will have no effect. The registers at 0x80, 0x88, 0x90, etc., are bit-addressable, all others are byte-addressable.

Special Function Registers (Generic 80515 SFRs)

Table 8 shows the location of the SFRs and the value they assume at reset or power-up.

Name	Location	Reset value	Description
P0	0x80	0xFF	Port 0
SP	0x81	0x07	Stack Pointer
DPL	0x82	0x00	Data Pointer Low 0
DPH	0x83	0x00	Data Pointer High 0
DPL1	0x84	0x00	Data Pointer Low 1
DPH1	0x85	0x00	Data Pointer High 1
WDTREL	0x86	0x00	Watchdog Timer Reload register
PCON	0x87	0x00	UART Speed Control
TCON	0x88	0x00	Timer/Counter Control
TMOD	0x89	0x00	Timer Mode Control
TL0	0x8A	0x00	Timer 0, low byte
TL1	0x8B	0x00	Timer 1, high byte
TH0	0x8C	0x00	Timer 0, low byte
TH1	0x8D	0x00	Timer 1, high byte
CKCON	0x8E	0x01	Clock Control (Stretch=1)
P1	0x90	0xFF	Port 1
DPS	0x92	0x00	Data Pointer select Register
S0CON	0x98	0x00	Serial Port 0, Control Register
S0BUF	0x99	0x00	Serial Port 0, Data Buffer
IEN2	0x9A	0x00	Interrupt Enable Register 2
S1CON	0x9B	0x00	Serial Port 1, Control Register
S1BUF	0x9C	0x00	Serial Port 1, Data Buffer
S1RELL	0x9D	0x00	Serial Port 1, Reload Register, low byte
P2	0xA0	0x00	Port 2
IEN0	0xA8	0x00	Interrupt Enable Register 0
IP0	0xA9	0x00	Interrupt Priority Register 0
S0RELL	0xAA	0xD9	Serial Port 0, Reload Register, low byte
P3	0xB0	0xFF	Port 3
IEN1	0xB8	0x00	Interrupt Enable Register 1
IP1	0xB9	0x00	Interrupt Priority Register 1
S0RELH	0xBA	0x03	Serial Port 0, Reload Register, high byte
S1RELH	0xBB	0x03	Serial Port 1, Reload Register, high byte
USR2	0xBF	0x00	User 2 Port, high address byte for MOVX@Ri
IRCON	0xC0	0x00	Interrupt Request Control Register
PSW	0xD0	0x00	Program Status Word
WDCON	0xD8	0x00	Baud Rate Control Register (only *WDCON*.7 bit used)
A	0xE0	0x00	Accumulator
B	0xF0	0x00	B Register

Table 8: Special Function Registers Reset Values

Accumulator (*ACC, A*): *ACC* is the accumulator register. Most instructions use the accumulator to hold the operand. The mnemonics for accumulator-specific instructions refer to accumulator as “A”, not *ACC*.

B **Register:** The *B* register is used during multiply and divide instructions. It can also be used as a scratch-pad register to hold temporary data.

Program Status Word (*PSW*):

MSB							LSB
CV	*AC*	*F0*	*RS1*	*RS*	*OV*	*-*	*P*

Table 9: *PSW* Register Flags

<table>
<tr><th>Bit</th><th>Symbol</th><th>Function</th></tr>
<tr><td><i>PSW.7</i></td><td><i>CV</i></td><td>Carry flag</td></tr>
<tr><td><i>PSW.6</i></td><td><i>AC</i></td><td>Auxiliary Carry flag for BCD operations</td></tr>
<tr><td><i>PSW.5</i></td><td><i>F0</i></td><td>General purpose Flag 0 available for user. <b>Not to be confused with the <i>F0</i> flag in the <i>CESTATUS</i> register.</b></td></tr>
<tr><td><i>PSW.4</i></td><td><i>RS1</i></td><td rowspan="2">Register bank select control bits. The contents of RS1 and RS0 select the working register bank:
<table>
<tr><th><i>RS1/RS0</i></th><th>Bank selected</th><th>Location</th></tr>
<tr><td>00</td><td>Bank 0</td><td>(0x00 – 0x07)</td></tr>
<tr><td>01</td><td>Bank 1</td><td>(0x08 – 0x0F)</td></tr>
<tr><td>10</td><td>Bank 2</td><td>(0x10 – 0x17)</td></tr>
<tr><td>11</td><td>Bank 3</td><td>(0x18 – 0x1F)</td></tr>
</table></td></tr>
<tr><td><i>PSW.3</i></td><td><i>RS0</i></td></tr>
<tr><td><i>PSW.2</i></td><td><i>OV</i></td><td>Overflow flag</td></tr>
<tr><td><i>PSW.1</i></td><td>-</td><td>User defined flag</td></tr>
<tr><td><i>PSW.0</i></td><td><i>P</i></td><td>Parity flag, affected by hardware to indicate odd / even number of “one” bits in the Accumulator, i.e. even parity.</td></tr>
</table>

Table 10: *PSW* bit functions

Stack Pointer (*SP*): The stack pointer is a 1-byte register initialized to 0x07 after reset. This register is incremented before PUSH and CALL instructions, causing the stack to begin at location 0x08.

Data Pointer: The data pointer (*DPTR*) is 2 bytes wide. The lower part is *DPL*, and the highest is *DPH*. It can be loaded as a 2-byte register (MOV DPTR,#data16) or as two registers (e.g. MOV DPL,#data8). It is generally used to access external code or data space (e.g. MOVC A,@A+DPTR or MOVX A,@DPTR respectively).

Program Counter: The program counter (*PC*) is 2 bytes wide initialized to 0x0000 after reset. This register is incremented during the fetching operation code or when operating on data from program memory.

Port Registers: The I/O ports are controlled by Special Function Registers *P0*, *P1*, and *P2*. The contents of the SFR can be observed on corresponding pins on the chip. Writing a '1' to any of the ports (see Table 11) causes the corresponding pin to be at high level (V3P3), and writing a '0' causes the corresponding pin to be held at low level (GND). The data direction registers *DIR0*, *DIR1*, and *DIR2* define individual pins as input or output pins (see section On-Chip Resources – DIO Ports for details).

Register	SFR Addres s	R/W	Description
P0	0x80	R/W	Register for port 0 read and write operations (pins DIO4…DIO7)
DIR0	0xA2	R/W	Data direction register for port 0. Setting a bit to 1 means that the corresponding pin is an output.
P1	0x90	R/W	Register for port 1 read and write operations (pins DIO8…DIO15)
DIR1	0x91	R/W	Data direction register for port 1.
P2	0xA0	R/W	Register for port 2 read and write operations (pins DIO16-DIO17)
DIR2	0xA1	R/W	Data direction register for port 2.

Table 11: Port Registers

All four ports on the chip are bi-directional. Each of them consists of a Latch (SFR '*P0*' to '*P3*'), an output driver, and an input buffer, therefore the MPU can output or read data through any of these ports. Even if a DIO pin is configured as an output, the state of the pin can still be read by the MPU, for example when counting pulses issued via DIO pins that are under CE control.

Special Function Registers Specific to the 71M6511

Table 12 shows the location and description of the 71M6511-specific SFRs.

Register	Alternative Name	SFR Addres s	R/W	Description
ERASE	*FLSH_ERASE*	0x94	W	This register is used to initiate either the Flash Mass Erase cycle or the Flash Page Erase cycle. Specific patterns are expected for *FLSH_ERASE* in order to initiate the appropriate Erase cycle (default = 0x00). 0x55 – Initiate Flash Page Erase cycle. Must be proceeded by a write to *FLSH_PGADR* @ SFR 0xB7. 0xAA – Initiate Flash Mass Erase cycle. Must be proceeded by a write to *FLSH_MEEN* @ SFR 0xB2 and the debug port must be enabled. Any other pattern written to FLSH_ERASE will have no effect.
PGADDR	*FLSH_PGADR*	0xB7	R/W	Flash Page Erase Address register containing the flash memory page address (page 0 thru 127) that will be erased during the Page Erase cycle (default = 0x00). Must be re-written for each new Page Erase cycle.
EEDATA		0x9E	R/W	I2C EEPROM interface data register
EECTRL		0x9F	R/W	I2C EEPROM interface control register. If the MPU wishes to write a byte of data to EEPROM, it places the data in *EEDATA* and then writes the 'Transmit' code to *EECTRL*. The write to *EECTRL* initiates the transmit sequence. See the section I2C Interface (EEPROM) for a description of the command and status bits available for *EECTRL*.

FLSHCRL		0xB2	R/W W R/W R	Bit 0 (*FLSH_PWE*): Program Write Enable: 0 – MOVX commands refer to XRAM Space, normal operation (default). 1 – MOVX @DPTR,A moves A to Program Space (flash) @ DPTR. This bit is automatically reset after each byte written to flash. Writes to this bit are inhibited when interrupts are enabled. Bit 1 (*FLSH_MEEN*): Mass Erase Enable: 0 – Mass Erase disabled (default). 1 – Mass Erase enabled. Must be re-written for each new Mass Erase cycle. Bit 6 (*SECURE*): Enables security provisions that prevent external reading of flash memory and CE program RAM. This bit is reset on chip reset and may only be set. Attempts to write zero are ignored. Bit 7 (*PREBOOT*): Indicates that the preboot sequence is active.
WDI		0xE8	R/W R/W W	**Only byte operations on the whole WDI register should be used when writing**. The byte must have all bits set except the bits that are to be cleared. The multi-purpose register *WDI* contains the following bits: Bit 0 (*IE_XFER*): XFER Interrupt Flag: This flag monitors the XFER_BUSY interrupt. It is set by hardware and must be cleared by the interrupt handler Bit 1 (*IE_RTC*): RTC Interrupt Flag: This flag monitors the RTC_1SEC interrupt. It is set by hardware and must be cleared by the interrupt handler Bit 7 (*WD_RST*): WD Timer Reset: The WDT is reset when a 1 is written to this bit.
INTBITS	*INT0…INT6*	0xF8	R	Interrupt inputs. The MPU may read these bits to see the input to external interrupts INT0, INT1, up to INT6. These bits do not have any memory and are primarily intended for debug use

Table 12: Special Function Registers

Instruction Set

All instructions of the generic 8051 microcontroller are supported. A complete list of the instruction set and of the associated op-codes is contained in the 651X Software User's Guide (SUG).

UART

The 71M6511 includes a UART (UART0) that can be programmed to communicate with a variety of AMR modules. A second UART (UART1) is connected to the optical port, as described in the optical port description.

The UART is a dedicated 2-wire serial interface, which can communicate with an external host processor at up to 38,400 bits/s ((with MPU clock = 1.2288MHz). The operation of each pin is as follows:

RX: Serial input data are applied at this pin. Conforming to RS-232 standard, the bytes are input LSB first. The voltage applied at RX must not exceed 3.6V.

TX: This pin is used to output the serial data. The bytes are output LSB first.

The 71M6511 has, several UART-related registers for the control and buffering of serial data.

A single SFR register serves as both the transmit buffer and receive buffer (*S0BUF,* SFR 0x99 for UART0 and *S1BUF,* SFR 0x9C for UART1). When written by the MPU, *SxBUF* acts as the transmit buffer, and when read by the MPU, it acts as the receive buffer. Writing data to the transmit buffer starts the transmission by the associated UART. Received data are available by reading from the receive buffer. Both UARTs can simultaneously transmit and receive data.

WDCON[7] selects whether timer 1 or the internal baud rate generator is used. All UART transfers are programmable for parity enable, parity, 2 stop bits/1 stop bit and XON/XOFF options for variable communication baud rates from 300 to 38400 bps. Table 13 shows how the baud rates are calculated. Table 14 shows the selectable UART operation modes.

	Using Timer 1	Using Internal Baud Rate Generator
Serial Interface 0	$2^{smod} * f_{CKMPU}/ (384 * (256\text{-}TH1))$	$2^{smod} * f_{CKMPU}/(64 * (2^{10}\text{-}S0REL))$
Serial Interface 1	N/A	$f_{CKMPU}/(32 * (2^{10}\text{-}S1REL))$

Note: *S0REL* and *S1REL* are 10-bit values derived by combining bits from the respective timer reload registers. *SMOD* is the *SMOD* bit in the SFR *PCON*. *TH1* is the high byte of timer 1.

Table 13: Baud Rate Generation

	UART 0	UART 1
Mode 0	N/A	Start bit, 8 data bits, parity, stop bit, variable baud rate (internal baud rate generator)
Mode 1	Start bit, 8 data bits, stop bit, variable baud rate (internal baud rate generator or timer 1)	Start bit, 8 data bits, stop bit, variable baud rate (internal baud rate generator)
Mode 2	Start bit, 8 data bits, parity, stop bit, fixed baud rate 1/32 or 1/64 of f_{CKMPU}	N/A
Mode 3	Start bit, 8 data bits, parity, stop bit, variable baud rate (internal baud rate generator or timer 1)	N/A

Table 14: UART Modes

Parity of serial data is available through the P flag of the accumulator. Seven-bit serial modes with parity, such as those used by the FLAG protocol, can be simulated by setting and reading bit 7 of 8-bit output data. Seven-bit serial modes without parity can be simulated by setting bit 7 to a constant 1. 8-bit serial modes with parity can be simulated by setting and reading the 9th bit, using the control bits TB80 (S0CON.3) and TB81 (S1CON.3) in the S0COn and S1CON SFRs for transmit and RB81 (S1CON.2) for receive operations. SM20 (S0CON.5) and SM21 (S1CON.5) can be used as handshake signals for inter-processor communication in multi-processor systems.

Serial Interface 0 Control Register (S0CON).

The function of the UART0 depends on the setting of the Serial Port Control Register S0CON.

MSB							LSB
SM0	*SM1*	*SM20*	*REN0*	*TB80*	*RB80*	*TI0*	*RI0*

Table 15: The *S0CON* Register

Serial Interface 1 Control Register (S1CON).

The function of the serial port depends on the setting of the Serial Port Control Register S1CON.

MSB LSB

SM	-	*SM21*	*REN1*	*TB81*	*RB81*	*TI1*	*RI1*

Table 16: The *S1CON* register

<table>
<tr><th>Bit</th><th>Symbol</th><th>Function</th></tr>
<tr><td><i>S0CON.7</i></td><td><i>SM0</i></td><td rowspan="2">These two bits set the UART0 mode:
<table>
<tr><th>Mode</th><th>Description</th><th>SM0</th><th>SM1</th></tr>
<tr><td>0</td><td>N/A</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>8-bit UART</td><td>0</td><td>1</td></tr>
<tr><td>2</td><td>9-bit UART</td><td>1</td><td>0</td></tr>
<tr><td>3</td><td>9-bit UART</td><td>1</td><td>1</td></tr>
</table></td></tr>
<tr><td><i>S0CON.6</i></td><td><i>SM1</i></td></tr>
<tr><td><i>S0CON.5</i></td><td><i>SM20</i></td><td>Enables the inter-processor communication feature.</td></tr>
<tr><td><i>S0CON.4</i></td><td><i>REN0</i></td><td>If set, enables serial reception. Cleared by software to disable reception.</td></tr>
<tr><td><i>S0CON.3</i></td><td><i>TB80</i></td><td>The 9th transmitted data bit in Modes 2 and 3. Set or cleared by the MPU, depending on the function it performs (parity check, multiprocessor communication etc.)</td></tr>
<tr><td><i>S0CON.2</i></td><td><i>RB80</i></td><td>In Modes 2 and 3 it is the 9th data bit received. In Mode 1, if <i>SM20</i> is 0, <i>RB80</i> is the stop bit. In Mode 0 this bit is not used. Must be cleared by software</td></tr>
<tr><td><i>S0CON.1</i></td><td><i>TI0</i></td><td>Transmit interrupt flag, set by hardware after completion of a serial transfer. Must be cleared by software.</td></tr>
<tr><td><i>S0CON.0</i></td><td><i>RI0</i></td><td>Receive interrupt flag, set by hardware after completion of a serial reception. Must be cleared by software</td></tr>
</table>

Table 17: The *S0CON* Bit Functions

Note: The speed in Mode 2 depends on the *SMOD* bit in the SFR *PCON*. See the *PCON* register description.

<table>
<tr><th>Bit</th><th>Symbol</th><th>Function</th></tr>
<tr><td><i>S1CON.7</i></td><td><i>SM</i></td><td>Sets the baud rate for UART1
<table>
<tr><th>SM</th><th>Mode</th><th>Description</th><th>Baud Rate</th></tr>
<tr><td>0</td><td>A</td><td>9-bit UART</td><td>variable</td></tr>
<tr><td>1</td><td>B</td><td>8-bit UART</td><td>variable</td></tr>
</table></td></tr>
<tr><td><i>S1CON.5</i></td><td><i>SM21</i></td><td>Enables the inter-processor communication feature.</td></tr>
<tr><td><i>S1CON.4</i></td><td><i>REN1</i></td><td>If set, enables serial reception. Cleared by software to disable reception.</td></tr>
<tr><td><i>S1CON.3</i></td><td><i>TB81</i></td><td>The 9th transmitted data bit in Mode A. Set or cleared by the MPU, depending on the function it performs (parity check, multiprocessor communication etc.)</td></tr>
<tr><td><i>S1CON.2</i></td><td><i>RB81</i></td><td>In Modes 2 and 3, it is the 9th data bit received. In Mode B, if <i>SM21</i> is 0, <i>RB81</i> is the stop bit. Must be cleared by software</td></tr>
<tr><td><i>S1CON.1</i></td><td><i>TI1</i></td><td>Transmit interrupt flag, set by hardware after completion of a serial transfer. Must be cleared by software.</td></tr>
<tr><td><i>S1CON.0</i></td><td><i>RI1</i></td><td>Receive interrupt flag, set by hardware after completion of a serial reception. Must be cleared by software</td></tr>
</table>

Table 18: The *S1CON* Bit Functions

Timers and Counters

The 80515 has two 16-bit timer/counter registers: Timer 0 and Timer 1. These registers can be configured for counter or timer operations.

In timer mode, the register is incremented every machine cycle meaning that it counts up after every 12 periods of the MPU clock signal.

In counter mode, the register is incremented when the falling edge is observed at the corresponding input signal *T0* or *T1* (*T0* and *T1* are the timer gating inputs derived from certain DIO pins, see the DIO Ports chapter). Since it takes 2 machine cycles to recognize a 1-to-0 event, the maximum input count rate is 1/2 of the oscillator frequency. There are no restrictions on the duty cycle, however to ensure proper recognition of 0 or 1 state, an input should be stable for at least 1 machine cycle.

Four operating modes can be selected for Timer 0 and Timer 1. Two Special Function Registers (*TMOD* and *TCON*) are used to select the appropriate mode.

Timer/Counter Mode Control Register (*TMOD*):

MSB							LSB
GATE	*C/T*	*M1*	*M0*	*GATE*	*C/T*	*M1*	*M0*
Timer 1				**Timer 0**			

Table 19: The *TMOD* Register

Bits *TR1* (*TCON.6*) and *TR0* (*TCON*.4) in the *TCON* register (see Table 22 and Table 23) start their associated timers when set.

Bit	Symbol	Function
TMOD.7 *TMOD.3*	*Gate*	If set, enables external gate control (pin int0 or int1 for Counter 0 or 1, respectively). When int0 or int1 is high, and *TRX* bit is set (see *TCON* register), a counter is incremented every falling edge on t0 or t1 input pin
TMOD.6 *TMOD.2*	*C/T*	Selects Timer or Counter operation. When set to 1, a Counter operation is performed. When cleared to 0, the corresponding register will function as a Timer.
TMOD.5 *TMOD.1*	*M1*	Selects the mode for Timer/Counter 0 or Timer/Counter 1, as shown in *TMOD* description.
TMOD.4 *TMOD.0*	*M0*	Selects the mode for Timer/Counter 0 or Timer/Counter 1, as shown in *TMOD* description.

Table 20: *TMOD* Register Bit Description

M1	*M0*	Mode	Function
0	0	Mode 0	13-bit Counter/Timer with 5 lower bits in the *TL0* or *TL1* register and the remaining 8 bits in the *TH0* or *TH1* register (for Timer 0 and Timer 1, respectively). The 3 high order bits of *TL0* and *TL1* are held at zero.
0	1	Mode 1	16-bit Counter/Timer.
1	0	Mode2	8-bit auto-reload Counter/Timer. The reload value is kept in *TH0* or *TH1*, while *TL0* or *TL1* is incremented every machine cycle. When *TL*(x) overflows, a value from *TH*(x) is copied to *TL*(x).
1	1	Mode3	If Timer 1 *M1* and *M0* bits are set to '1', Timer 1 stops. If Timer 0 *M1* and *M0* bits are set to '1', Timer 0 acts as two independent 8-bit Timer/Counters.

Table 21: Timers/Counters Mode Description

Note: *TL0* is affected by *TR0* and gate control bits, and sets *TF0* flag on overflow.
TH0 is affected by *TR1* bit, and sets *TF1* flag on overflow.

Timer/Counter Control Register (*TCON*)

MSB LSB

TF1	*TR1*	*TF0*	*TR0*	*IE1*	*IT1*	*IE0*	*IT0*

Table 22: The *TCON* Register