

PowerPC 403GC Implementation of IR Remote Control

December 4, 1995

John McCardle

IBM Microelectronics
Dept D95/Bldg 060
3039 Cornwallis Road
Research Triangle Park, NC 27709
E-Mail: jmccardle@vnet.ibm.com

ABSTRACT

The focus of this paper describes how to use the advanced integration of the IBM PowerPC 403GC to minimize system costs in a set top box design.

The role of the set top in the media market is to enable consumption of digital/analog video content from a network service provider. Set top systems must meet consumer price points to be effective enablers. IBM has developed a PowerPC embedded controller that provides a high degree of component integration and meets the performance levels required to achieve a lower cost interactive solution.

This paper discusses the advantages of IBM's PowerPC 403GC in a set top box design. An overview of the IBM PPC403GC microprocessor is given, then methods are proposed for implementing an IR Remote and IR Blaster without external digital hardware.

INTRODUCTION

The PowerPC 403GC embedded controller is a 32-bit RISC processor integrated with a memory management unit, DRAM controller, memory mapped I/O controller, DMA unit, serial port, interrupt controller and JTAG port. The design is fully static with maximum operating frequencies of 25 and 33 Mhz.

The RISC core has 4 separate functional units: the execution unit (EU), memory management unit (MMU), cache unit and timer unit. The EU contains the ALU, the load/store unit, thirty-two 32-bit general purpose registers and a host of special purpose registers. The execution unit is connected to the cache unit through the MMU. The MMU has a fully associative 64 entry transition lookaside buffer (TLB) and provides enhanced features over PowerPC 6xx MMUs making it more suitable for embedded use. In particular, the 403GC MMU provides variable page sizes rather than a fixed 4K page size

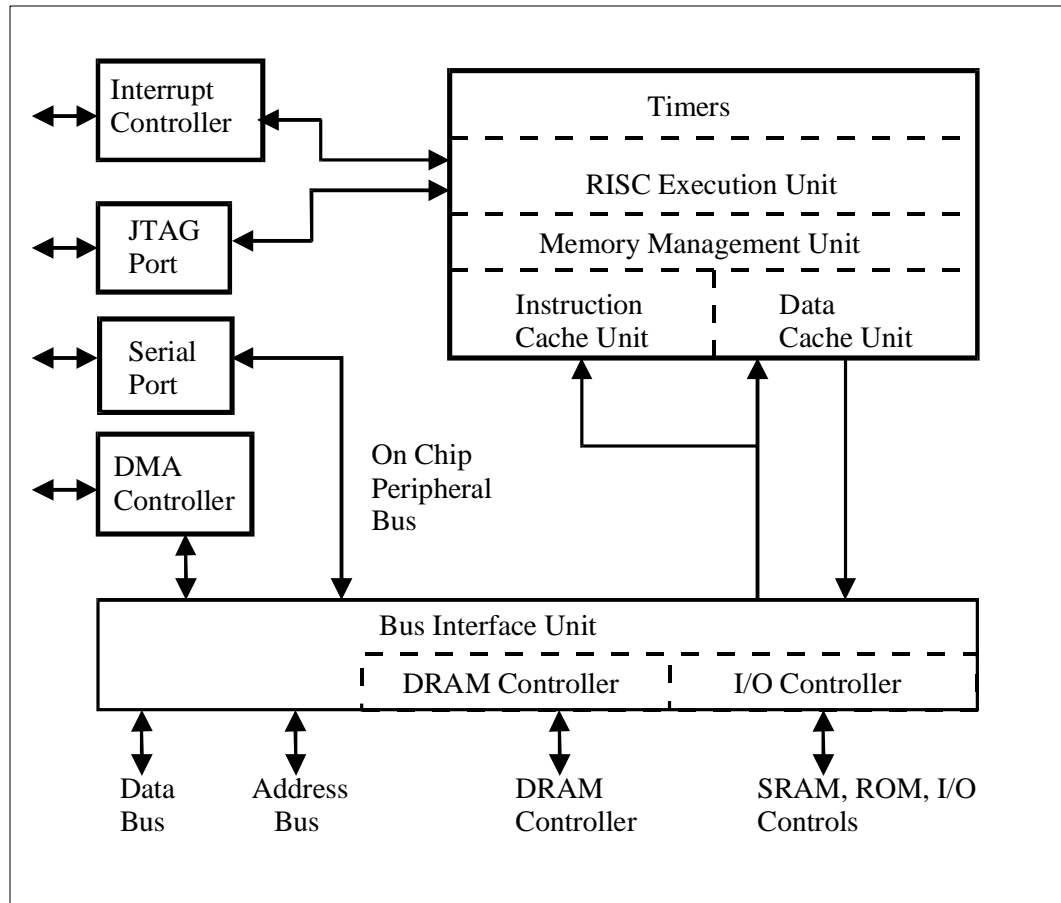


Figure 1. PowerPC 403GC

in the 6xx. The range of page sizes is 1K to 16 MBytes in multiples of 4. The programmer is able to mix page sizes in the TLB. For example, entry 1 may point to a 1K page and entry 2 points to a 16M page. Another unique feature of the 403GC MMU is software control of page replacement. This feature allows the RTOS to effectively lock entries in the TLB, thus eliminating page switching latency for timing critical code.

The cache unit contains 2 Kbytes of instruction cache and 1 Kbyte of data cache. Both caches are two-way set associative. The timer unit contains a 64-bit free running timer that runs at either the system clock or an external frequency. There are 3 functions based around the free running timer: the programmed interval timer (PIT), the fixed interval timer (FIT) and a watch dog timer (WDT). The PIT is a programmable decrementing counter which generates an interrupt when it reaches zero. The FIT generates interrupts in repeating periods. The WDT enables recovery from hung software by generating an interrupt if a bit in the timer status register is not maintained.

Surrounding the core is a highly integrated bus interface unit, an interrupt controller, an external bus master arbiter, a serial port and a JTAG port. The bus interface unit (BIU) consists of an external bus interface (Address, Data, Byte Enable, R/W, etc.), a DRAM controller and a memory mapped I/O controller (MMIO). Chip selects 0:3 are used to control SRAM, ROM or I/O devices. The MMIO chip selects are programmable and can

access up to 64 Mbytes of address space. The programmer determines CS, OE and WBE assertion and hold time. Burst devices are also supported.

Chip selects 4:7 may individually control DRAM modules or SRAM/ROM/IO devices. Each chip select is fully programmable and can access up to 64 Mbytes of address space. In DRAM mode the programmer controls RAS/CAS assertion, hold and pre-charge timing. In fast page mode, typical access times are 2-1-1-1-1 and 3-2-2-2-1 depending on processor clock frequency and memory speed. In SRAM mode, chip selects 4:7 function exactly as chip selects 0:3.

The DMA unit has 4 channels which may be operate in fly-by, memory-to-memory or buffered mode. In all modes, the memory controller generates the required control lines for the transfer. Bursting is allowed in fly-by mode and chaining is provided in fly-by and buffered modes.

Six external interrupt pins are provided, five general purpose and one critical. The general purpose interrupts are level or edge sensitive while the critical interrupt pin is falling edge active.

The serial port controls transmitted data, received data and data flow control functions. The maximum speed is 1/16 of the system clock. An external serial clock input is also provided. The JTAG port is used primarily for hardware debuggers. It provides an interface for executing control commands such as Reset, Halt, Run and Load. It also allows access to internal registers and cache.

SET TOP CONTROL

One of the primary functions of the CPU in a set top environment is generation of control codes to the transport stream demultiplexer, MPEG Video/Audio decoder, graphics processor and the network interface module. Since these commands are dispatched by the CPU, it is logical (and cost effective) that the command signals be received and decoded by the CPU. The primary source of command signals is an IR remote control. The basic function of control commands are channel selection, cursor control, service menu, field selection in an electronic program guide, etc. Optional commands are play, stop, pause, reverse and forward for interactive video via the network interface or for control of external devices such as digital video disks and standard VCR's via an IR Blaster.

IR REMOTE CONTROL

A remote control function can be implemented for a pulse width modulated (PWM) or pulse position modulated (PPM) signal using the programmable edge triggering of the 403GC general purpose external interrupt pins. This technique is desirable because decoding the pulse modulated signal in software allows flexibility in the selection of the modulation method (PWM or PPM) and pulse width requirements for command representation.

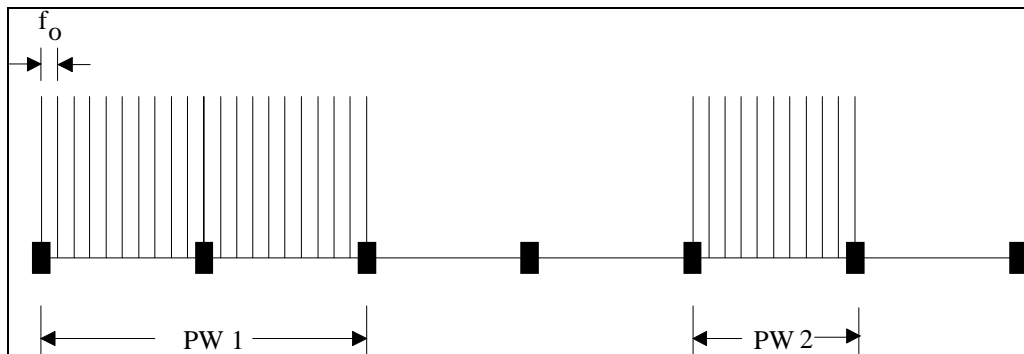


Figure 2: Pulse Width Modulated Signal

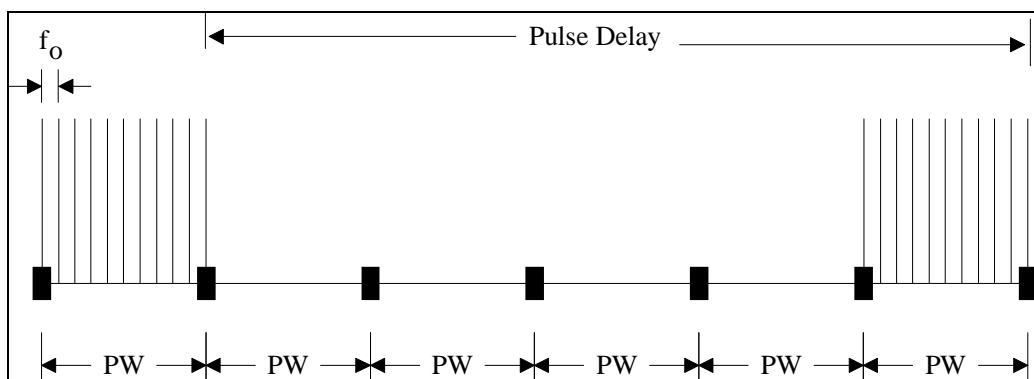


Figure 3: Pulse Position Modulated Signal

A typical PWM signal is shown in figure 2. Light is modulated by a carrier frequency f_o , typically in the range of 30-50 KHz. The light pulse is modulated to increase the immunity to noise produced by modern fluorescent lighting. PW1 and PW2 are specified pulse widths, each representing an encoded command. For this example the following values are assigned:

- $f_o = 36.7 \text{ KHz}$
- $PW_{\min} = 560 \mu\text{s}$
- $PW_{\min \text{ separation}} = 1.125 \text{ ms}$

and the infrared light has a wavelength $\lambda = 940 \text{ nm}$.

A typical PPM command pulse is shown in figure 3. The value f_o is the carrier frequency as in the PWM signal. However, in PPM, the pulse width is fixed and the pulse delay is the encoded command character. The following values are assigned:

- $f_o = 36.7 \text{ KHz}$
- $PW = 560 \mu\text{s}$
- $D_{\min} = 1.125 \text{ ms}$

Where D_{\min} is minimum delay between the two pulses and the infrared light wavelength $\lambda = 940 \text{ nm}$. In order for the CPU to recognize the signal, the light pulses must be translated into a square wave signal. A typical

converter circuit is shown in figure 4. These types of signal converters are commonly available. For example, the Sharp GP1U582Y satisfies the above requirements while also providing some fluorescent rejection. The output from the pulse shaping circuit is routed to one of the general purpose interrupt pins on the 403GC.

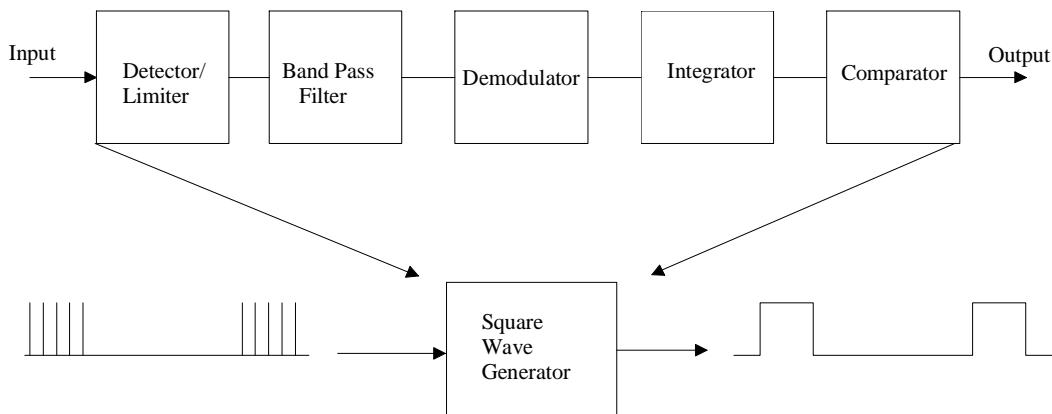


Figure 4: Square Wave Generator

The basic format for decoding a pulsed command signal requires the edge triggering capability of the general purpose interrupt pin to detect the beginning and end of the command signal and the 403 programmable interval timer to determine the time interval and provide some error detection. The PIT is a 32-bit counter which decrements at the frequency of the 64-bit time base (either the sysclock or an external frequency less than half the sysclock). At the beginning of the command signal, the PIT is loaded with an initial value and decrements until the end of the command is detected. The time interval is derived from the remaining value in the PIT and the requested command is determined from this value. To generate error recovery, the initial value loaded in the PIT is greater than the longest expected delay of the pulsed command. Thus, when the PIT times out, an error has occurred and recovery is provided by the PIT interrupt subroutine (ISR).

The process for decoding PWM and PPM follows the basic framework, but the algorithmic implementations are slightly different. The PWM case is simpler than PPM so I will attack it first. The idea is to measure the time interval between the rising and falling edges of the pulse, then translate it into a specific command. The following pseudo code illustrates the process.

PWM Interpreter

```
init()
edge=positive
PIT_init = max_width
end

ISR()
if (edge == positive)
    PIT = PIT_init
    edge = negative
else
    interval = PIT_init - PIT_value
    if (interval > min_PW)
        cmd = interpret(interval)
    edge = positive
end
```

In PPM the command signal is decoded by measuring the time interval between the rising or falling edge of two pulses. I assume falling edge detection in this example. The PPM case is a bit more complex because I do not know whether the detected edge is from a begin or end pulse. The following pseudo code describes the PPM process.

PPM Interpreter

```
init()
edge=negative
PIT_init = max_interval
begin_flag = 1
end

ISR()
if (begin_flag == 1)
    PIT = PIT_init
    begin_flag = 0
else
    interval = PIT_init - PIT_value
    if (interval > min_PW)
        cmd = interpret(interval)
        begin_flag = 1
    end
```

The begin_flag is used to keep track of which pulse is occurring. If a pulse is detected and no other pulse occurs, the PIT will time out and the corresponding ISR will recover. The condition “interval > min_PW” provides some noise immunity. Alternatively, one can interpret the time interval between the falling edge of the initial pulse and the rising edge of the terminal pulse. This allows the rising/falling edge selector bit in the IOCR to act as the ‘begin_flag’.

Both the PPM and PWM algorithms use a routine called *interpret(interval)* to translate the interval length to a command code. This routine is implementation specific. I specified the time intervals, which represent specific commands, to be multiples of the minimum pulse width. Thus command 3 will correspond to a time interval of $3 \cdot PW_{\min} = 3 \cdot 560 \mu s = 1.68 \text{ ms}$. At a 27 Mhz clock rate, this translates to 45360 decrements. It is tempting to divide this value by the number of ticks in 1 pulse width, 15120 @ 27 Mhz, to obtain the command value. This technique assumes no margin of error in the pulse shaping circuitry, so it is not practical. A second technique is to subtract 1/2 a pulse width (1-1/2 in PPM since a 2 pulse width minimum interval is recommended) from the interval, then divide by the number of ticks in a pulse width. Since it is binary division, the resultant will be an integer corresponding to the command number. This technique produces results that are less dependent upon pulse shaping circuitry. The technique does require the minimum pulse width emitted by the remote to be within a tolerance of $\pm (\text{min_PW}) / 2$ distributed over the total number of commands. One way of satisfying this requirement is to send a calibration pulse when the system is “turned on” by the remote. A set top is not totally powered down unless supply voltage is removed. Thus a status flag can be maintained by the processor to indicate whether video is being displayed or not (user’s perception of “powered down”). The CPU will determine whether to measure or decode the IR pulse based upon the setting of a power status flag.

The interpreted commands are dispatched to the appropriate modules to accomplish a particular function. Certain commands may be generated to control an external VCR. The next section describes a process through which the 403GC serial port may be used to accomplish an IR Blaster function.

IR BLASTER

A PWM or PPM can be generated from the 403GC using the pattern generation mode (PGM) feature of the serial port. When bit 7 of the serial port transmitter command register is set, the start and stop bits are not added to the 8-bit character prior to transmission. Thus, one or several characters (strung together) can generate one PWM or PPM command. The baud rate generator is programmed such that one bit corresponds to the minimum pulse width. This is the formula for determining baud rate:

$$\text{Baud Rate} = \text{Clock} \div [16 * (\text{BRD} + 1)]$$

where BRD is two 8-bit registers, BRDH and BRDL, concatenated together to form the baud rate divisor (BRD). For example, using a 27 Mhz clock source, programming the BRDH to 0x3h and the BRDL to 0xB0h will produce a pulse width of 560 μ s as in the received PPM and PWM signals above. This series of transmitted values:

0x11111111b
0x11000000b



will produce a PWM command code of 10. This series of transmitted values:

0x10000000b
0x00000000b
0x00000010b



will produce a PPM command code of 21 (0x1010000b is command code 1 due to the 2*PW minimum separation). Note that the value of the last bit is maintained in between transmissions so no false terminations will occur. In order for the processor not to be burdened with polling, the TxReady interrupt is enabled prior to transmission. The processor will be interrupted each time the transmit buffer becomes empty. The key feature is programmability. The pulse width and pulse separation are controlled by the baud rate generator and transmitted patterns.

IrDA TRANSMITTER

Pattern generation mode can also be used to transmit IrDA signals. An IrDA 1.0 signal is similar to RS232 in that it has a start bit, 8 data bits and 1 or 2 stop bits. The IrDA signal differs in that the polarity is switched and the transmitted pulse width is a fraction of the data rate. Figure 5 shows a unipolar RS232 signal and the corresponding IrDA signal. The RS232 pulse width 'T' is the inverse of the baud rate and the IrDA pulse width 'P' is specified as $T * 3/16$. Note that a '0' is represented by an IR pulse and a '1' is represented by no pulse.

To generate IrDA using the 403GC serial port, the baud rate generator is programmed for 16 times the IrDA data rate. For instance, a "transmission" rate of 9600 baud requires a value of 0x000Ah in the BRG (27 Mhz sysclock) to produce a 153.4 Kbaud "generation" rate. A single bit of data is transmitted by sending 2 consecutive bytes to the transmit buffer. A transmitted value is represented by 0xe000h for logical zero and 0x0000 for logical one. The data stream in figure 5 is represented by the following bytes (in hexadecimal notation):

e000 0000 0000 e000 0000 e000 0000 0e00 0000 0000

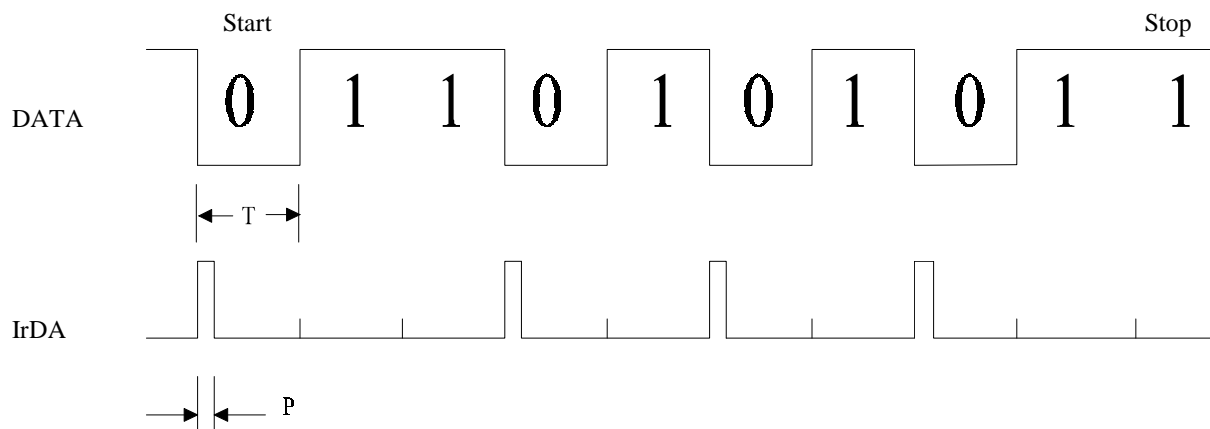


Figure 5: IrDA Signal Format

In the above example, a 27 Mhz sysclk is used. This frequency does not generate exactly 9600 baud. The actual transmission rate is 9588 baud, which produces a 0.12% error. For this frequency, transmission rates less than or equal to 9600 baud have a 0.12% error and rates above 9600 have a 8.5% error. However, IrDA data rates of 57.6K, 19.2K, 9.6K, 4.8K and 2.4K baud can be generated exactly by connecting a 14.745 Mhz external oscillator to the 403GC serclk input frequency pin. Note

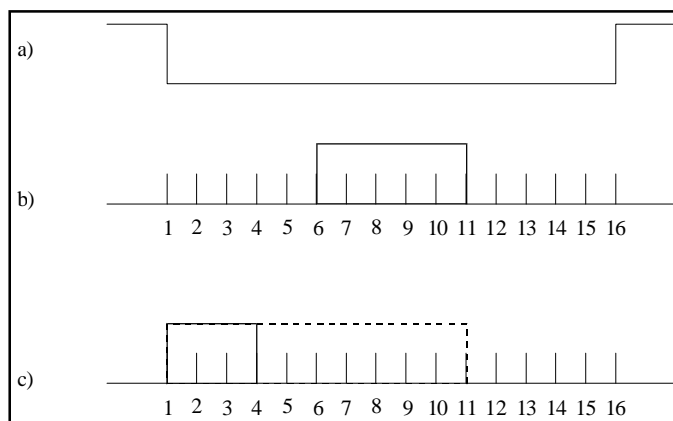


Figure 6: a) One bit of serial data. b) Serial port receive sampling time. c) Received IrDA pulse.

that the serial clock is constrained to be less than half the frequency of the sysclk. Thus a 14.745 Mhz serclk rate requires the sysclk frequency to be greater than 29.45 Mhz. This is well within the 33 Mhz maximum operating frequency of the 403GC. The XmitD pin is connected to an infrared transmitter such as the IBM31T1100 Integrated Transceiver Module which easily handles all the above data rates.

It is not possible to receive IrDA signals with the 403GC serial port without external hardware. The 403GC receiver samples the RecvD pin at 16 times the data rate. Figure 6a shows a logical zero data bit. From figure 6b, the five samples at the mid-point of the incoming pulse are used to determine the value for the data bit. External hardware is required to latch the incoming data and extend it past the receiver sampling window (figure 6c).

Alternatively, the detected IrDA signal is connected to a general purpose interrupt pin. The same techniques used in remote control detection are applied to determine the transmitted data. The IrDA signal is detected by the IBM31T1100, inverted and sent to the interrupt pin (figure 7). The dual edge

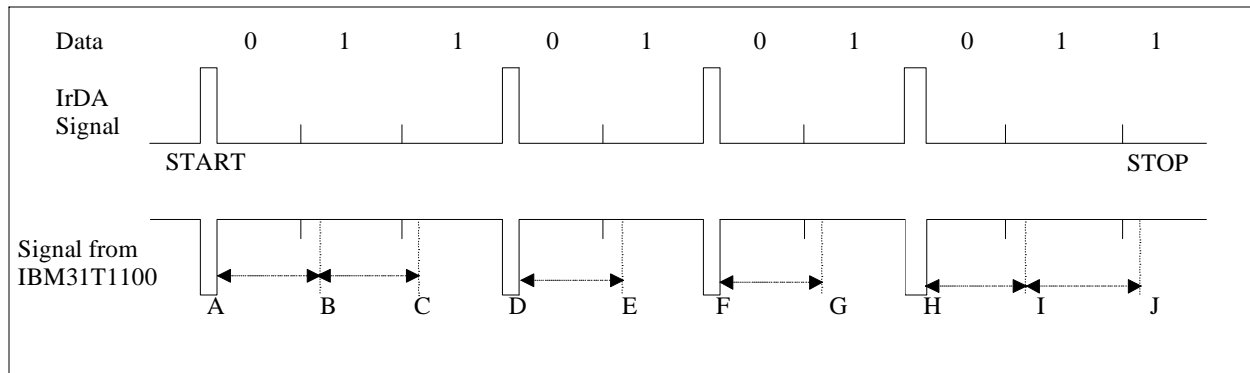


Figure 7: IrDA Receiver using interrupt pin.

triggering feature is used to detect and measure an incoming pulse (measurement is necessary for noise rejection). At the falling edge of a valid start pulse, the PIT is loaded with a value corresponding to the data rate (A in figure 7). If no pulse is received the PIT times out, automatically reloads itself (if enabled) and generates an interrupt (B in figure 7). The interrupt handler updates a GPR to indicate that a '1' was received. If another pulse is detected before the PIT expires (D in figure 7), the pulse is measured and if valid, a general purpose register is updated to indicate that a '0' was received. A counter is updated whenever a bit is detected. When the terminal count is reached, a stop bit is expected. If a stop bit occurs, one byte of data is available to system software. If no stop bit occurs, an error code is sent.

The IrDA 1.0 data rates are specified up to 115.2 Kbaud. High data rates will have an impact on system performance. Using the techniques described above to implement an IrDA transceiver puts a practical limit on the data rate of 9600 baud. During transmission, interrupts are generated every 52 μ s. While receiving, an interrupt is generated in the range of 55 μ s to 95 μ s with an average of 69.5 μ s. With an average interrupt latency of 15 μ s at the beginning and end of the ISR, there is 20 μ s left to execute the handler. At 33 Mhz this corresponds to approximately 660 instructions, if running from cache, or 28 cache line (four 32-bit words) loads or stores with 70 ns DRAM.

CONCLUSION

Designers can exploit the integrated features of the PowerPC 403GC to reduce system cost in set top designs. An IR remote control signal can be detected using the programmable edge triggering of interrupt pins. Decoding is accomplished in software which provides the flexibility to support PWM or PPM signals. An IR Blaster can be accomplished using the pattern generation mode feature of the serial port. An IrDA transceiver can be accomplished using pattern generation mode for transmission and edge detecting of the interrupt pins for reception.

IBM will continue to enhance products and services as new technologies emerge. Therefore, IBM reserves the right to make changes to its products, other product information, and this publication without prior notice. Please contact your local IBM Microelectronics representative on specific standard configurations and options.

IBM assumes no responsibility or liability for any use of the information contained herein. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of IBM or third parties. NO WARRANTIES OF ANY KIND, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE ARE OFFERED IN THIS DOCUMENT.

