APPLICATION NOTE

BENCHMARK ST72 vs. PIC16

by Microcontroller Division Application Team

ABSTRACT

This document presents the results of a competitive analysis between the STMicroelectonics ST72254 and the Microchip PIC16F876. These two microcontrollers (MCUs) have been chosen for comparison because they are in a similar performance category and were introduced on the market at the same time.

The comparison of the two MCUs is divided into two major parts. First the cores, with a comparison of their architecture including performance benchmarks. These benchmarks are based on assembler and C routines that are representative of typical microcontroller applications. The second part examines the peripherals in terms of their functionality and to what extent they off-load the core and the driver software.

Finally, you will find a table summarizing the weak and the strong points of each MCU.

Two files are appended to this document, you can find them in our Web server (mcu.st.com) in the application note section. The first one entitled “Performance comparison between ST72254 and PIC16F876” includes the results given in this document plus the description of the source and the compilation options used. This file was created in order to allow you to easily reproduce the benchmark. The second file regroups all the source files used.

The information on the PIC16F876 is based on the Microchip datasheet: DS30292A.PDF

Table of Contents

1 DEVICE DESCRIPTION

Table 1. Microchip

	PIC16C62B	PIC16C72A	PIC16F873	PIC16F876
Program memory	2K*14	2K*14	4K*14 (Flash)	8K*14 (Flash)
RAM	128*8	128*8	192*8 128*8 EEPROM	368*8 128*8 EEPROM
Stack	8*13-bit, can store up to 8 addresses			
CPU Frequency	Up to 5MHz (with 20MHz oscillator)			
Oscillator	RC / Ceramic / Crystal			
Power saving	One mode (Sleep Mode)			
Operating Range	0°C to +70°C or -40°C to +85°C (optional -40°C to +125°C on 16C62B or 16C72A but not on 16F87x)			
Package	SO28/PDIP28 (windowed version available for 16C62B or 16C72A)			
I/O port	22 pins with individual direction control. Output are push-pull (except 1 true open-drain pin). 8 pins may have internal pull-up (globally selected). Current up to 25mA.			
Watchdog	On-chip RC and a 8-bit prescaler (the prescaler is shared with TIMER0). The time-out period may vary between 7ms and 4.2s. But due to the internal RC variations, the time-out period is between 896ms and 4.2s for the same settings.			
Timer	8-bit timer/counter with 8-bit prescaler (shared with WDT) 16-bit timer/counter with prescaler, internal/external clock and possible dedicated oscillator. 8-bit timer with 8-bit period register, prescaler and postscaler (this timer does not support external clock despite what is described in the Microchip datasheet)			
Serial communication	SPI, I²C	SPI, I²C	SPI/I²C,SCI	
ADC	No	8-bit with 5 inputs	10-bit with 5 inputs	
LVD	Selectable, one level (known as BOR in the Microchip documentation)			
RESET	WDT, POR, LVD, External. During internal reset (WDT, LVD or POR) the reset state is not externally visible)			
ISP	M_{CLR},Clk,Data (need 12V on M_{CLR})		Two modes: The old one, and a new +5V only mode (it needs one more pin)	

Table 2. STMicroelectronics

	ST72104G1	ST72104G2	ST72216G1	ST72215G2	ST72254G1	ST72254G2
Program memory	4K*8 (flash)	8K*8 (flash)	4K*8 (flash)	8K*8 (flash)	4K*8 (flash)	8K*8 (flash)
RAM	256*8	256*8	256*8	256*8	256*8	256*8
Stack	128*8	128*8	128*8	128*8	128*8	128*8
Oscillator	RC / Ceramic / Crystal / internal / Clock security system (clk filter, safe oscillator, limitation detection)					
Power saving	Four power saving modes					
CPU Frequency	Up to 8MHz (with 16MHz oscillator)					
Operating Range	-40°C to +85°C (optional -40°C to +125°C)					
Package	SO28/SDIP32					
I/O port	22 pins with individual direction control and individual mode control (push-pull, open-drain, input with or without pull up). Open drain mode is limited to logic level except for PA4 and PA6, which are true open drain pins.					
Watchdog	64 values selectable for time-out from 1.5ms to 98ms @ Fcpu=8MHz					
Timer	One 16-bit timer/counter with 2 IC, 2 OC, 1 PWM			One 16-bit timer/counter with 2 IC, 2 OC, 1 PWM One 16-bit timer with 2 IC, 2 OC, 1 PWM		
Serial communication	SPI			SPI	SPI, I²C	SPI, I²C
ADC	No	No	8-bit with 6 inputs			
LVD	3 selectable levels					
RESET	WDT, POR, LVD, External. During internal reset (WDT, LVD or POR), the reset state is externally visible (reset pin in low state). Internal pull-up.					
ISP	Reset, ISPSEL, IPSCLK, IPSDATA					

2 CORE

2.1 ARCHITECTURE

• PIC16

The PIC16 family is based on a Harvard architecture (i.e.: data and program memory spaces are separated). These processors have a RISC core (*Reduced Instruction Set Computer*) with only 35 instructions. All these instructions have the same size: 14 bits (for the PIC16 family). They are executed in one CPU cycle except branches which need two cycles.

The processor uses an accumulator called W (Working register).

Despite its Harvard architecture, the PIC16F87x program memory is **readable and writable by the MCU** during program execution. But the access method is not simple. This means, it is not efficient to store data tables directly in program memory.

• ST72

The ST72 family is based on a Von Neuman architecture (i.e.: data and program share the same memory space). These processors have a CISC core (*Complex Instruction Set Computer*) with 63 instructions. These instructions are from 2 to 4 bytes long.

The processor uses an accumulator called A.

Read access throughout the entire memory space is very easy. In consequence, the program memory can be efficiently used to store constant tables.

2.2 RAM

Table 3. RAM

ST7xxG1-2	256*8 with no bank switching
PIC16C62/72A	128*8 accessible through **2 banks** of 128 bytes (the banks contain 96 bytes of user data and the hardware registers)
PIC16F873	192*8 accessible through **2 banks** of 128 bytes (the banks contain 96 bytes of user data and the hardware registers)
PIC16F876	368*8 accessible through **4 banks** of 128 bytes (the banks contain 96 bytes of user data and the hardware registers)

On the ST72, the first 128 bytes of the RAM array is call the zero page (addresses from $80 to $FF). The data in this zone can be accessed in short addressing mode (8-bit addresses) reducing the opcode size and speeding-up the access. This is not a bank system, because it is possible (but slower) to access the entire RAM array using the long addressing mode (16-bit addresses).

2.3 ROM

Table 4. ROM

ST72254G1	4 K*8 (-> up to 2048 instructions) FLASH which can store either programs or data
ST72254G2	8 K*8 (-> up to 4096 instructions) FLASH which can store either programs or data
PIC16C62/72A	2K*14 (->really 2048 instructions) OTP or EPROM (UV erasable)
PIC16F87x	8K*14 (->really 8092 instructions) FLASH for program storage 256*8 EEPROM for data storage. This EEPROM is accessible only through an indexed mode **(9 instructions are needed to read one EEPROM Byte)**

• Program memory access

The PIC16F87x allows the program to read or write program memory during normal execution. 14 opcodes are needed to read one word (14-bit) and 23 opcodes to write one word. The MCU is halted during a write (10ms) and after it restarts with the next instruction.

In the ST72, read access to the program memory is very easy, but there is no write access allowed during execution.

• Conclusion

The two program memory organisations are quite different. A PIC16 with 8K*14 of memory could be compared to a ST72 with 16K*8 (which does not exist in 28 pins). But due to the RISC instruction set of the PIC16, more assembly instructions are needed to code the same program in the ST72.

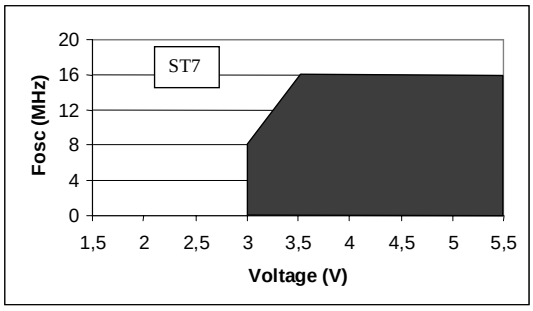
2.4 FREQUENCY

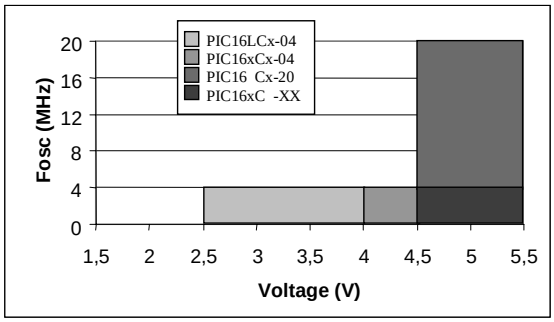
Table 5. Frequency

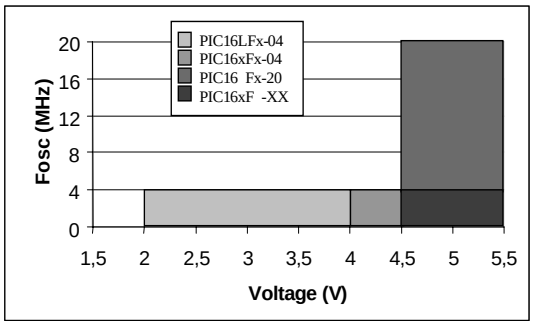
	PIC16C62 / PIC16F87x	ST72254
$F_{oscilator}$ max	20Mhz	16Mhz
F_{CPU} max	5MHz (Fosc/4)	8MHz (Fosc/2)
$F_{INSTRUCTION}$ max	5MHz (Instructions are 1 or 2 cycles long)	4MHz (quickest instructions are two cycles long)

2.5 VOLTAGE RANGE

PIC16C62/72A	4.5 V - 5.5 V
PIC16LC62/72A	2.5 V - 5.5 V
PIC16F87x	4.5 V - 5.5 V
PIC16LF87x	2.0 V - 5.5 V
ST72254	3.0 V - 5.5 V







• Conclusion

The PIC16F87x has a wider voltage range. It can work at 2.0 V

But to take advantage of this voltage range, the operating frequency must be lowered to 4MHz. Moreover, two different MCUs are needed to cover the entire voltage range.

For example: at 3.5V, the ST72254 can work at full speed (16MHz) and the PIC can run only at 4MHz.

2.6 STACK

• PIC16

The stack is only **8 levels deep** (it is like a 16 bytes deep stack) and is separated from the user RAM. The stack cannot be accessed by software (no PUSH or POP!).

• ST72

The stack is up to 128 bytes deep, it can store **64 addresses.** It shares the user RAM.

The stack can be easily accessed by software (PUSH/POP instructions, Stack Pointer is R/W)

2.7 REGISTERS

Table 6. Registers

	PIC16	ST72
Accumulator	W (8-bit)	A (8-bit)
Program counter	13-bit (memory is organised in words of 14-bit)	16-bit (memory is organised in bytes)
Stack Pointer	not accessible	7-bit
Index Register	FSR (8-bit) **needs bank switching**	Two Index Register X,Y: 8-bit

2.7.1 CODE CONDITION REGISTER vs. STATUS REGISTER

The two registers are nearly the same but the ST72 has a negative bit to indicate if the result of the last operation is negative or not.

2.8 ADDRESSING MODE

The PIC16 offers only 3 addressing modes (Immediate, Relative, Indexed). The RAM is organised in banks of 96 bytes. This means that bank switching is needed to access the entire RAM. The architecture of the **PIC does not allow programs to directly address FLASH or EEPROM**.

The **ST72 is really more powerful**. It has 11 different addressing modes. And it never needs bank switching. And, last but not least, it allows programs to access RAM or FLASH (for reading) in exactly the same way.

2.8.1 Indexed mode

The **PIC16 has one index register** (FSR), but it works like a data memory location. This means that it is not possible to work directly with this register. The only operations which can be done without using the accumulator are: incrementation, decrementation and comparison with zero.

Consequently, **table manipulations are quite hard to handle**.

The **ST72 has two index registers**. The following operations on this register can be done directly (without using the accumulator): incrementation, decrementation, **comparison with a value and loading a value (literal or stored in a register)**.

2.9 INSTRUCTION SET

Due to its RISC architecture the **PIC16**'s instruction set is very small, only **35 instructions**. But these instructions are executed quickly (all need only one CPU cycle except branches which need two cycles).

The internal clock is divided by 4. In consequence, compared to the ST72 (where the clock is divided by 2) the instructions are like 2 or 4 cycle instructions of the ST72.

The **ST72** has a CISC architecture with **63 instructions** (nearly twice as many as the PIC). These instructions are one to four bytes long and need between 2 to 12 CPU cycles.

The **ST72** has a **multiplication** instruction 8-bit*8-bit, result on 16-bits. This multiplication needs only 11 CPU cycles. To do the same with PIC architecture, you need at least 37 CPU cycles and 35*14 bits of program memory (or 71 CPU cycles and 16*14 bits of memory).

2.10 INTERRUPT

Table 7. Interrupt sources/vectors

	PIC16C62B	PIC16C72A	PIC16F87x	ST72254
IT Sources	7	8	13	19
IT Vectors	1	1	1	**7**
Register saved	No	No	No	**PC, X, A and CC**

Table 8. Interrupt reaction time

	PIC16		ST72	
	Min	**Max**	**Min**	**Max**
$T_{INSTRUCTION}$*	1	2	2	12
T_{JUMP}	2	2	10	10
$T_{CONTEXT\ SAVING}$	4	**10**	Included in the jump	
TOTAL (T_{CPU})	7	14	10	22
Interrupt reaction time	**1.4 µs**	2.8 µs	**1.25 µs**	2.75 µs

*$T_{INSTRUCTION}$ is the number of f_{CPU} cycles needed to complete the current instruction

• PIC16

During an interrupt, only the return address is automatically saved onto the stack. Saving the context needs to be done manually (10 instructions needed) and the interrupt sub-routine must also restore the context (6 instructions needed).

• ST72

The ST72 has 7 different interrupt vectors, it allows you to easily create independent interrupt subroutines. Context saving is done automatically. The only weak point of the interrupt mechanism of the ST72 is that the interrupt priority is fixed by hardware. But some ST72s have a Nested Interrupt feature (ex: ST72311R6)

• Conclusion

The interrupt response times are very much the same. But for marketing reasons, Microchip does not take the time needed to save the context into account. This is why Microchip announces better response times. Moreover, the **PIC architecture provides only one interrupt vector.** So, the interrupt sub-routine has to read all the interrupt flags in order to find which interrupt needs to be serviced.This means a significant software overhead when many interrupt sources are used.

2.11 POWER SAVING MODE

• PIC16

The PIC16 has **only one power saving mode:** the sleep mode. In this mode the oscillator is shut off. Any of the following events can cause a wake-up from sleep mode:

- External reset
- Watchdog Timer (in sleep mode a Watchdog time-out does not reset the MCU)
- Interrupt (if the individual interrupt enable bit is set)
- TMR1 interrupt (if the TIMER1 clock is an external clock or if it is it's own oscillator)
- Capture mode interrupt
- Special event trigger (cf. A/D and TIMER1)
- SPI or I²C in slave mode
- USART in slave mode (if available)
- A/D conversion if A/D clk is RC
- EEPROM write complete

The MCU needs 1024 T_{osc} (256 CPU cycles) to wake-up (except in RC mode where the wake-up is immediate)

• ST72

The ST72 has **three different power saving modes**:

- **Slow mode**, which allows the internal clock of the device to be reduced (4 different speeds available)
- **Wait mode**, which turns off the CPU but keeps the clock active for the peripherals. It can be exited by any peripheral interrupt (ex: timer, SPI...). Wake-up from this mode is immediate.
- **Halt mode**, which turns off the CPU and the clock system. It can be exited by external interrupt. The CPU needs 4096 CPU cycles (8092 T_{osc}) to wake-up in order to stabilise the oscillator.

• Conclusion

With its three power saving modes, the **ST72 is quite flexible**. It is possible to adjust the power consumption to the exact needs of the application.

However, in situations when the PIC16 sleep mode can be used, it might be more efficient. This is because serial communication is still possible in slave mode. In terms of current consumption, the PIC16's sleep mode is not unlike the ST72's halt mode.

Moreover, the PIC16's wake-up is faster: 1024 oscillator cycles compared to 8092 for the ST72, and with an RC oscillator the PIC16 is woken-up immediately but the ST72 is not.

2.12 POWER CONSUMPTION DATA (TAKEN FROM THE DATASHEET)

Table 9. PIC16 power consumption data (Low power device)

PIC16LF87x, PIC16LC62B/72A low consumption devices	PIC16LF87x		PIC16LC62B/72A	
	Typ	Max	Typ	Max
Run mode OCS1=external square wave from rail to rail. All I/O tristated, pulled to V_{DD}. /M_{CLR} pulled to V_{DD}. V_{DD}=3.0V				
XT mode, RC oscillator 4MHz (Fcpu=1MHz)*	2mA	3.8mA	2mA	3.8mA
LP mode, Fosc=32KHz (Fcpu=9KHz) WDT disabled	20uA	48uA	22.5uA	48uA
Power down current (sleep mode), all I/O in high impedance and tied to V_{DD} or V_{SS}. V_{DD}=3.0V				
WDT enabled, -40°C to +85°C	7.5uA	30uA	7.5uA	30uA
WDT disabled, -40°C to +85°C	0.9uA	5uA	0.9uA	5uA
WDT disabled, 0°C to +70°C	0.9uA	5uA	0.9uA	5uA
LVD add typ 85uA max 200uA of consumption (characterized but not tested) (Vdd=5V) TIMER1 oscillator add approx 20uA (Vdd=5V)				

*In RC mode the current through R is not included.

Table 10. PIC16 power consumption data (standard device)

PIC16F87, PIC16C62B/72A	PIC16F87x		PIC16C62B/72A	
	Typ	Max	Typ	Max
Run mode OCS1=external square wave from rail to rail. All I/O tristated, pulled to V_{DD}. /M_{CLR} pulled to V_{DD}. V_{DD}=5.5V				
XT mode, RC oscillator 4MHz (Fcpu=1MHz)*	2mA	5mA	2.7mA	5mA
HS mode, Fosc=20MHz (Fcpu=5Mhz)	**10mA**	20mA	10mA	20mA
Power down current (sleep mode), all I/O in high impedance and tied to V_{DD} or V_{SS}. V_{DD}=4.0V				
WDT enabled, -40°C to +85°C	10.5uA	42uA	10.5uA	42uA
WDT disabled, -40°C to +85°C	1.5uA	19uA	1.5uA	19uA
WDT disabled, 0°C to +70°C	1.5uA	16uA	1.5uA	16uA
WDT disabled, -40°C to +125°C	2.5uA	19uA	2.5uA	19uA
LVD add typ 85uA max 200uA of consumption (characterized but not tested) TIMER1 oscillator add approx 20uA				

*In RC mode the current through R is not included.

Table 11. ST72254 power consumption data

ST72254	Typ	Max
Run mode OCS1=external square wave. All I/Os in input mode with a static value V_{DD} or V_{SS}. CPU running with memory access.		
Fosc=16MHz (Fcpu=8Mhz) V_{DD}=5V	**5.5mA**	10mA
Fosc= 8MHz (Fcpu=4Mhz) V_{DD}=5V	3mA	6mA
Slow mode. All I/Os in input mode with a static V_{DD} or V_{SS} value		
Fosc=16MHz (Fcpu=500Khz) V_{DD}=5V	0.7mA	1.4mA
Fosc= 8MHz (Fcpu=250Khz) V_{DD}=5V	0.5mA	1mA
Wait mode. All I/Os in input mode with a static V_{DD} or V_{SS} value (values are characterized but not tested)		
Fosc=16MHz (Fcpu=8Mhz) V_{DD}=5V	2mA	4mA
Fosc= 8MHz (Fcpu=4Mhz) V_{DD}=5V	1mA	2mA
Slow Wait mode. All I/Os in input mode with a static V_{DD} or V_{SS} value		
Fosc=16MHz (Fcpu=500Khz) V_{DD}=5V	0.4mA	0.8mA
Fosc= 8MHz (Fcpu=250Khz) V_{DD}=5V	0.2mA	0.4mA
Halt mode. All I/Os in input mode with a static V_{DD} or V_{SS} value. LVD disabled		
	0.5uA	5uA

• Conclusion

In run mode, **the ST72254 draws less power** than the PIC16F876. 5.5mA versus 10mA and with the maximum values the difference is even bigger 10mA vs. 20mA. For the power saving modes, the conclusion is hard to establish because the modes available are quite different.

3 CORE PERFORMANCE COMPARISON

STMicroelectronics has developed two sets of test routines related to 8-bit and low-end 16-bit microcontroller applications to evaluate the **computing performance** of **microcontroller cores**. These routines have been implemented on ST72254 and PIC16F876 Microcontroller Units.

- The first set of routines has been written in **assembler language** to optimize their implementation and focus on core performance, without being dependent upon compiler code transformation.
- The second set tries to evaluate the performance of the two MCUs and their respective **C compilers**. This benchmark uses a C language program, representative of an automotive application. The C compilers used were from **Hiware** on the ST72 and from **Hi-Tech** on the PIC16.

The speed of the two MCUs has been compared in two ways:

- Firstly, at the maximum frequency commercially available on each MCU. this means at an external frequency of 16MHz on the ST72 and of 20MHz on the PIC16.
- Secondly, at the same current consumption level (10mA).

Table 12. Current Consumption data (taken from datasheets)

	F_{ext}	F_{CPU}	Consumption (Max)	
ST72254	**16MHz**	8MHz	**10 mA**	Run mode
PIC16C72A	20MHz	5MHz	20 mA	Run mode
PIC16C72A	**10MHz***	2.5MHz	**10 mA**	Run mode

* this value is determined by interpolation

As you can see, to reach the same power consumption level on the two MCUs, the PIC's running frequency must be lowered to 10Mhz (ext.) and the ST72 can keep its maximum frequency of 16MHz (ext.).

3.1 ASSEMBLER TEST ROUTINES OVERVIEW

The set of test routines is made of 8 assembly programs which cover all the typical needs of an MCU application. This routine tests only the core performance and does not include any peripheral management.

Table 13. Assembler test routine overview

Abbreviated name	Full name	Description	Features stressed
string	String search	search a 16-byte string in a 128-character array in ROM	8-bit data block manipulation string manipulation
char	Character search	search a byte in a 40-byte array in ROM	8-bit data manipulation char manipulation
bubble	Bubble sort	sort of a one-dimension array of 10 16-bit integers	16-bit data manipulation integer manipulation
blkmov	Block move	move a 64-byte block from a place in RAM to another	8-bit data block manipulation block move
convert	Block translation	translate a 80-byte block in a different format	8-bit data manipulation use of a lookup table
shright	16-bit value right shift	shift a 16-bit value five places to the right	16-bit data manipulation bit manipulation
bitsrt	Bit manipulation	set, reset, and test of 3 bits in a 128-bit array	bit computation bit and 8-bit data manipulation
32div	32-bit by 16-bit division	Unsigned division of a 32-bit dividend by a 16-bit divisor	bit manipulation 16-bit substraction
16mul	16-bit integer multiplication	multiplication of two unsigned words giving a 32-bit result	16-bit data computation integer manipulation

• Notes on memory accesses used in the test routines:

The size of the arrays manipulated by the test routines has been chosen in order to minimize RAM bank switching on the PIC16 processor. This means that **the results do not include any overhead for memory bank switching on the PIC16** MCU. But with the complexity-levels of real-world applications, the paginated memory can be a major source of time and code overhead.

For the same reason on the ST72 the data are placed in the zero page, allowing to use the short addressing mode.

3.2 ASSEMBLER TEST RESULTS

Table 14. Execution Speed

	At Maximum Frequency			At a Current of 10mA			Description
	PIC16 @20MHz	ST72 @16MHz	Speed Ratio ST72/PIC16*	PIC16 @10MHz	ST72 @16MHz	Speed Ratio ST72/PIC16*	
string	371 µs	282 µs	1.32	741 µs	282 µs	2.63	search a 16-byte string in a 128-character array in ROM
char	84 µs	57 µs	1.49	169 µs	57 µs	2.98	search a byte in a 40-byte array in ROM
bubble	752 µs	857 µs	0.88	1504 µs	857 µs	1.76	sort of a one-dimensional array of 10 16-bit integers
blkmov	154 µs	121 µs	1.28	308 µs	121 µs	2.55	move a 64-byte block from one place in RAM to another
convert	256 µs	241 µs	1.06	513 µs	241 µs	2.13	translate a 80-byte block into a different format
shright	6 µs	10 µs	0.65	13 µs	10 µs	1.29	shift a 16-bit value five places to the right
bitsrt	36 µs	62 µs	0.58	72 µs	62 µs	1.17	set, reset, and test of 3 bits in a 128-bit array
32div	124 µs	222 µs	0.56	248 µs	222 µs	1.12	Unsigned division of a 32-bit dividend by a 16-bit divisor
16mul	41 µs	18 µs	2.29	72 µs	18 µs	4.58	multiplication of two unsigned words giving a 32-bit result
AVG.	**203 µs**	**208 µs**	**0.98**	**406 µs**	**208 µs**	**1.95**	**Average results**

*The speed ratio is calculated as follows: (Time PIC16)/(Time ST72).
So, a number higher than 1 means that the ST72 is faster.

- **At their maximum frequency**, we can see that **the execution speed of the two MCUs is truly comparable**. Even though, in this configuration the external frequency of the PIC16 is higher (20Mhz) than that of the ST72 (16MHz).
- **At the same power consumption** level, the ST72254 is significantly better. In other words, for the same power consumption budget, **the ST72254 is nearly 2 times more powerful than the PIC16F87x**.

3.3 RESULTS ANALYSIS

• Bit manipulation:

The Microchip architecture is very fast for bit manipulation. Modifying one bit in a data byte can be performed in only 1 CPU cycle which means in 0.2µs@20MHz. To do the same operation the ST72 needs 5 CPU cycles (0.625µs@16MHz).

• Memory access:

The PIC16 is faster in direct addressing mode (1 cycle 0.2µs@20MHz compared to 3 cycles 0.375µs@16MHz). This is very useful for many parts of the application where variables are directly used, for example in loop control.

The indirect mode of the PIC16 (needed for table or string manipulation) is very slow, because its index register cannot be loaded without losing the content of the accumulator.

The ST72 with its two index registers and its wider choice of addressing modes allows very easy (and fast) data manipulation.

To summarize, when there is no need for indirect addressing, the PIC16 runs faster. But for more complex algorithms the ST72 is better (faster and easier to program).

Moreover, the test routines use only a small amount memory so, there is no problem of bank switching. But in a real (and large-sized) application, this could be a major limitation of the PIC16 architecture.

• Use of Constant tables

The PIC16 cannot read the content of its ROM directly, this means that, constant tables must be handled in a strange way. It needs a sub-routine call and a computed jump. This takes at least 6 CPU cycles (1.2µs@20MHz). On the ST72, the same operation needs 6 CPU cycles (0.625µs@16MHz).

Moreover, due to the need for a 8-bit computed jump, reading a constant table bigger than 256 bytes needs more time and code on the Microchip architecture.

• Multiplication:

Here, the comparison is easy: The ST72 has a 8-bit multiply instruction. The PIC16 does not have any. Doing multiplication entirely by software implies considerable time and code overhead.

3.4 C TEST ROUTINES OVERVIEW

The source program has been provided by a customer. It has 9 modules controlled by the main routine in 'FILE7.C'. It uses all instructions usually found in C language programs.

The source makes heavy use of **unsigned char, bit and table manipulations.** The modules are described in the following table. We have tried to highlight the main features of each module.

Table 15. Module description

Module	#lines	Description	Features stressed
file1	204	after evaluation of a data by a switch, manipulation of "global" data and function calls (~100)	switch/case processing function calls
file2	538	definition of functions with data manipulation, for loop, while statement, if and switch uses	loop statements arithmetic computation
file3	93	definitions of 6 functions manipulating arrays, one function doing intensive calculation	array manipulation bit calculation
file4	251	mainly load of constant tables, and manipulation of structures at the end	constant table manipulation structure use
file5	164	exactly the same file than file8.c but using switch/case statement	switch/case statement
file6	68	if processing and bitwise computation	bit manipulation and if use
file7	133	the file contains the "main", initialises data and calls functions in the other files	function calls data initialisation
file8	88	exactly the same file then file5.c	if/else statements
file9	34	signed char data computation	signed data manipulation

Note: Number of lines: 1918.

3.4.1 Modification of the source files

The PIC16 data memory is organised in banks which contains up to 96 bytes of RAM. But, the Hi-Tech C compiler (PIC16) does not distribute the variables automatically into these different banks. So, to make the compilation phase work, **the sources files need to be modified**. Two files have been modified: 'FILE2.C' and 'FILE2.H'. The modifications consist of using the key-words `bank1` and `bank2` to place some of the variables in the different memory banks.

For the **ST72** we have made two sets of sources. The first one called **standard does not have any modifications**. The second one called **improved** is modified to take more advantage of the use of the zero page. The only modified file is 'FILE2.H' where two lines are added:

```
#pragma DATA_SEG SHORT ZEROPAGE
#pragma DATA_SEG DEFAULT
```

In order to make heavily-used global variables directly accessible (in short addressing mode)

3.5 C TEST RESULTS

Table 16. Results

	ST72 (Standard)	ST72 (improved)*	PIC16*
ROM usage	7242 bytes	6849 bytes	5529 words of 14 bits
RAM usage	200 bytes	200 bytes	180 bytes
Execution time			
CPU cycle	96374	94312	62609
Time at the maximum frequency	12.04ms@16Mhz	**11.78ms@16Mhz**	**12.52ms@20Mhz**
Time at a 10mA current	12.04ms@16Mhz	11.78ms@16Mhz	**25.04ms@10Mhz**

* These source files have been modified

- In terms of execution speed, the results are the same as in the assembly routine test: **the execution speed of the two MCUs is truly comparable at their maximum frequency**.
  And for the **same power consumption budget, the ST72 is nearly 2 times more powerful** than the PIC16.

- In terms of program size: the reported size is between 20% and 30% higher on the ST72 than on the PIC16. But this is normal, because the memory of the PIC16 is organised in words of 14 bits and not in bytes (8-bit) like on the ST72.

4 PERIPHERALS

4.1 I/O PORTS

Both PIC16 and ST72 propose I/O ports with individual direction control. Both have 22 I/O pins.

• PIC16F87x

- Port A (6 pins): analog input. TTL buffers except RA4 which has Schmitt trigger input or open drain output (true open drain but limited to 8.4 V)
- Port B (8 pins): TTL buffer, input with **pull-up (globally selectable)**
  RB0 external interrupt pin.
  RB4->RB7: Interrupt when a value on a pin changes **(selectable globally** on input pins).
- Port C: 8 pins with Schmitt trigger input buffer, TTL output

Note: The interrupt-on-change is dedicated to interfacing a 4*4 keypad. It can be used to wake-up the CPU when a key is pressed. But if the interrupt occurs while a read is in progress on PORTB (even a bit set on a pin) the interrupt may not be taken into account. Consequently, **if the interrupt-on-change feature is used, reading PORTB must be avoided**.

• ST72254

For all I/O pins it is possible to **individually choose** between :

- floating input
- pull-up input with interrupt
- push pull output
- open drain output. The open drain outputs are limited to logic level. But there are 2 true open-drain pins.

Table 17. Output Voltage level

	PIC16	ST72 standard output	ST72 High sink output
Vol	**0.6 V** / 8.5mA	0.5 V / 2 mA,**1.3 V** / 5mA	**0.5 V** / 8mA, 1.3V / 20mA
Voh	V_{DD}-**0.7 V** / 3 mA	V_{DD}-**0.8 V** / 2mA ,V_{DD}-2 V / 5mA	V_{DD}-**0.8 V** / 2mA ,V_{DD}-2 V / 5mA

PIC16F87x: All the TTL buffers can Source/Sink high current
ST7254: has 8 outputs able to sink high current.

• Conclusion

The ST72 offers more flexibility in the I/O port configuration but the PIC16 outputs can sink or source bigger currents.

4.2 CLOCK

Table 18. Clock characteristics

	PIC16C62B/72A		PIC16F87x		ST72254	
	Min	**Max**	**Min**	**Max**	**Min**	**Max**
External clock	0 MHz	20 MHz	0 MHz	20 MHz	0	16 MHz
Quartz/ ceramic oscillator	0.1 MHz	20 MHz		20 MHz	1 MHz	16 Mhz
RC oscillator	0 MHz	4 MHz			1 MHz	14 Mhz
External R	3 K	100 K	3 K	100 K	22 K	47 K
External C	20 pF		20 pF		0	470 pF
Internal RC	No		No		**Typ: 4 Mhz**	
Safe clock oscillator	No		No		**250 kHz**	**430 kHz**
Clock spikes filter	No		No		**Yes with detection**	

The ST72 has a useful clock module. It ensures that the MCU will always run, even if the crystal has a problem. This security system can be a welcome feature for a lot of applications.The ST72 proposes also an internal RC oscillator which allows to reduce the need for external components to the minimum.

4.3 TIMER

Table 19. Timer summary

PIC16F87	ST72254
2 timers 8-bit	
1 timer 16-bit	2 timers 16-bit
2 PWM (same frequency)	2 PWM and 2 IC
or 2 IC (same time base)	or 1 PWM and 2 OC and 3 IC
or 2 OC (same time base)	or 2 One Pulse Mode and 2 IC and 2 OC
or 1 PWM and 1 IC	or 4 IC and 4 OC
No simple way to access the 16-bit timer	**security system** (latches the LSB after reading the MSB)
(12 instructions to read the timer)	

Table 20. PWM

	PIC (Fosc=20MHz)	ST7 (Fosc=16MHz)
Maximum Resolution	10-bit	16-bit
Possible frequencies at full resolution	3	3
Maximum frequency at full resolution	19.53KHz	61Hz
Maximum frequency at 10-bit resolution	19.53KHz	3.9KHz
Resolution available at 1 kHz	1024	8000
Resolution available at 20kHz	1000	200

• PIC

It has 3 different timers but with some restrictions.

- TIMER0 (8-bit) share its prescaler with the Watchdog.
  The prescaler allows to choose between 8 different timer frequencies.
  Internal or External clock
  (if the prescaler is used to increase the Watchdog time-out there is no possibility to choose the TIMER0 frequency.)
- TIMER1 (16-bit) has four prescaler values.
  Internal or External clock (in asynchronous mode or in synchronous mode).
  Dedicated oscillator (up to 200KHz).
- TIMER2 (8-bit) has three prescaler values.
  Period register, in fact it is a compare register and when a match is detected, the timer is cleared.

CCP module:

It is a register which can be configured as an **input capture** register **or** as an **output compare** register **or** as a **PWM** duty cycle register (**it is an exclusive or!**).

It is possible to automatically clear the timer when there is an output compare match (this possibility is referenced in the Microchip documentation as Special-event Trigger).

- The **PIC16C62B** has **one CCP** module so it is possible to do either one IC or one OC or one PWM.
- The **PIC16C72A** also seems to have only **one CCP** module but there are some mistakes in the Microchip documentation.
- The **PIC16F87x** has **two CCP** modules so it is possible to do two PWMs or two ICs or OCs or to do a mix. The Special event Trigger of the second CCP module can clear TIMER1 and start an A/D conversion.

The two CCP modules share the same timer resources, and there are also some major restrictions due to the interaction between the two modules.

Table 21. CCP module interaction (PIC16F87x)

CCPx mode	CCPy mode	Restriction
Capture	Capture	Same time base
Capture	Compare	Same time base (The special event Trigger clears the timer)
Compare	Compare	Same time base (The special event Trigger clears the timer)
PWM	PWM	**Same frequency!** and same update rate
PWM	Capture	no problem
PWM	Compare	no problem

The PIC16 does not provide any security system for ensuring the integrity of reading or writing the 16-bit timer, which needs two 8-bit accesses. To secure the access to the CCP register (IC, OC or PWM), Microchip proposes to first shut off the function of the CCP register then to do the access and finally to re-enable the CCP module.

To directly read or write the 16-bit timer, Microchip proposes the following sequence of code

Table 22. Example of code for accessing the 16-bit timer

Example 12-3: Writing a 16-bit Free Running Timer

```
; All interrupts are disabled
CLRF TMR1L ; Clear Low byte, Ensures no
; rollover into TMR1H
MOVLW HI_BYTE ; Value to load into TMR1H
MOVWF TMR1H, F ; Write High byte
MOVLW LO_BYTE ; Value to load into TMR1L
MOVWF TMR1H, F ; Write Low byte
; Re-enable the Interrupt (if required)
CONTINUE ; Continue with your code
```

Table 23. Example of code for accessing the 16-bit timer

Example 12-4: Reading a 16-bit Free Running Timer

```
; All interrupts are disabled
MOVF TMR1H, W ; Read high byte
MOVWF TMPH ;
MOVF TMR1L, W ; Read low byte
MOVWF TMPL ;
MOVF TMR1H, W ; Read high byte
SUBWF TMPH, W ; Sub 1st read with 2nd read
BTFSC STATUS,Z ; Is result = 0
GOTO CONTINUE ; Good 16-bit read
;
; TMR1L may have rolled over between the read of the high and
low bytes.
; Reading the high and low bytes now will read a good value.
;
MOVF TMR1H, W ; Read high byte
MOVWF TMPH ;
MOVF TMR1L, W ; Read low byte
MOVWF TMPL ;
; Re-enable the Interrupt (if required)
CONTINUE ; Continue with your code
```

This program is taken from a Microchip application note.

Finally, reading the timer requires between 9 and 12 CPU cycles and 12 14-bit words of program memory

• ST72

The ST72 has one or two 16-bit timers.

The first timer (non-optional) can work with an external clock but the second one (optional) cannot.

The two timers are really independent. The timer value cannot be set directly (just a clear is possible).

Each timer has

- A prescaler which provides 3 different time bases
- 2 input capture functions
- 2 output compare functions
- 1 PWM (use the 2 OCs)
- 1 One pulse mode (use 1 IC and 1 OC)
- 4 alternate functions on the I/O ports (IC1,IC2,OC1,OC2). The first timer may also use an external clock pin

On the ST72 with 2 16-bit timers, there are 4 ICs, 4 OCs, 2 PWMs (really independent) and 2 One-pulse modes.

The ST72 offers a simple security system for accessing the 16-bit register (Timer, IC or OC): The access to the MSB part disables the function or latches the LSB value until the access to the LSB part is performed.

• Conclusion

The **PIC16 offers a better PWM for frequencies over 3.9KHz**.

At frequencies below this limit, the precision of the ST72 PWM is better.

Moreover, **the ST72 offers two really independent PWMs** while the PIC16 PWMs are linked and must run at the same frequency.

Microchip has three timers, this would appear to be better than on the ST72. But there are a lot of restrictions in the use of these timers.

This means that with its two timers the **ST72 is more powerful**. In fact, **it depends on the exact needs of the application**. If the application can deal with the interactions between the different modules, then it can take an advantage of the three timers available on the PIC16.

4.4 WDT: WATCHDOG TIMER

Table 24. Watchdog timer

	PIC16	ST72
Timer source	Dedicated RC oscillator	Internal clock
Min Time out	Between 7-30 ms (depending of the device)	1.5ms (16MHz osc.)
Max Time out	Between 896ms-4.2s (depending of the device)	98.3ms (16MHz osc.)
Possible values	8 values using the TIMER0 prescaler	64
Refresh method	Clear the timer	Can load any value in the timer
Power down mode	The WDT can wake-up the CPU from sleep mode	The WDT is stopped in halt mode

• PIC16

It uses an internal dedicated RC oscillator. **It shares an 8-bit prescaler** with the TIMER0 module. So when using the prescaler to increase the time-out period of the Watchdog, there **is no possibility to change the TIMER0** (8-bit timer) frequency.

The time-out period is between 7 and 33 ms without prescaler (**the variations are due to frequency variations of the RC oscillator between each device**).

Using the prescaler, **8 time-out** values can be chosen giving a time-out period up to 7*128=896 ms or 33*128=4.2s regardless of Fcpu.

In sleep mode, the Watchdog is still running. A Watchdog time-out while the MCU is in sleep mode will wake-up the MCU (and not reset it). So this feature prevents erroneously putting the CPU in sleep mode. If the CPU is put back in sleep mode, the Watchdog is automatically cleared. The WDT can only be cleared. So you cannot easily modify the time-out period during execution.

• ST72

The Watchdog timer is completely independent of the other timers. The time-out period can be chosen between **64 values**. It goes from 1.5ms to 98.3 ms (with a 16Mhz oscillator)

In Halt mode, the oscillator is shut off, so the Watchdog is also shut off. It restarts when the MCU is woken-up. You can choose the value you want to load into the WDT. So you can easily choose the WDT time-out for the next block of instructions.

• Conclusion

The Watchdog of the ST72 allows the time-out period to be defined more precisely. But, the Watchdog of the PIC16 offers a protection against erroneously entering sleep mode.

4.5 LVD: LOW VOLTAGE DETECTOR

Table 25. Low Voltage detector

	PIC16		ST72	
	Min	**Max**	**Typ**	**Max**
Selectable LVD	Yes (one level)		Yes (3 levels)	
Reset release threshold	3.7 V	4.3 V	4.3 V (high) 3.9 V (med) 3.35 V (low)	4.5V(high) 4.05V (med) 3.45 V (low)
Reset generation threshold	3.7 V	4.3 V	3.85 V(high) 3.50 V (med) 3.00 V (low)	4.25 V (high) 3.80 V (med) 3.20 V (low)
Hystereris	Done by holding in reset state for a minimum time after V_{DD} rises again		250 mV	

• PIC16

It offers a single level low voltage detector. The CPU is put in Reset, when the voltage goes below the low-level value for more than 100 µs (min time to detect the low voltage). When V_{DD} rises-up again, the CPU enters power-up reset mode. The power-up Reset is active for at least 28ms. If V_{DD} goes down again during this time then the power-up timer is cleared.

• Conclusion

The LVD module of the **PIC16 is limited** because it can only be used in 5V operating mode. But if the application runs at 5V there is no significant difference between the two modules.

4.6 ADC: ANALOG TO DIGITAL CONVERTER

The A/D conversion modules of the PIC16 and of the ST72 are based on the same A/D conversion method (successive approximations). The PIC16C62B does not have any ADC cell.

Table 26. ADC characteristics

	PIC16C72A	PIC16F87	ST72254
Resolution	8-bit	**10-bit**	8-bit
Channel	5	5	6
Conversion time (under a 10K source impedance)	20 µs	20 µs	**3µs**
Maximum conversion frequency	50KHz	50KHz	333KHz
Voltage reference (full scale)	V_{DD} or V_{REF} (**I/O pin**)		V_{DD}
Total absolute error	+-1 lsb	+-1 lsb	+-1 lsb
Integral linearity error	+-1 lsb	+-1 lsb	+-0.5 lsb
Differential linearity error	+-1 lsb	+-1 lsb	+-0.5 lsb
Full scale error	+-1 lsb	+-1 lsb	+-0.5 lsb
Offset error	+-1 lsb	+-1 lsb	+-0.5 lsb

Test condition:

Pic : At V_{REF}=V_{DD}=5.12V

ST7: V_{DD}=5V worst-case temperature, negative injection V_{DD}=V_{DDA}=5V fcpu=8MHz F_{ADC}=8MHz

All these values are guaranteed (maximum value).

4.6.1 Features

The **PIC16F87x** offers the possibility to assign an **I/O pin to be the high analog voltage reference**. This is not available on 28-pin versions of the ST72.

The **PIC16F87x ADC can work when the MCU is in sleep mode**, using the timer output compare function to start the A/D conversion. The timer is automatically cleared. When the conversion is ready, the MCU is woken-up. To provide the A/D clock, there is a **built-in RC** oscillator (using the RC oscillator, the conversion time is 48 µs for the PIC16F87). This feature provides a way to do A/D conversion at a fixed rate with a minimum software overhead.

On the ST72, this mechanism doesn't exist. But, in wait mode the A/D module is still able to work, so an output compare interrupt can be set to wake-up the MCU and then it can read the last converted value, and set the OC up again to finally go back in wait. The software overhead is slightly more significant.

• Conclusion

The A/D module of the **ST72 is really better than the module of the PIC16C72A** (faster and more precise). **But in comparison with the PIC16F87, the conclusion is more difficult to establish**. The PIC16F87 A/D module has a better resolution and the automatic sample rate feature, but the ST72 A/D module runs significantly faster. So the advantages depend on the application.

4.7 SPI SERIAL COMMUNICATION

• Common Features

- Clk edge and polarity select
- Interrupt on transfer complete
- Selectable baud rate
- MSB first transmission
- Single buffered transmission register
- Double buffered reception register
- $\overline{SS}$ allows to select the active slave

• PIC16

- **Receive overflow detection** (in slave mode).
- Can **work while in sleep** (slave mode only), the MCU is woken-up by the IT on transfer complete.
- The SPI and I²C cells of the PIC share the same registers, so **it is not possible to use the SPI and the I²C simultaneously.**

• ST72

- Write collision detection: if a write is made to the register before the transfer is complete.
- Mode Fault detection: if the $\overline{SS}$ is pulled low while in master mode, it automatically switches from master to slave mode.
- Can work in wait mode (master or slave mode), the MCU is woken-up by the interrupt on transfer complete. But it can't work in Halt mode.

Table 27. Baud Rate

	PIC16 (F_{ext}=20MHz)	ST72 (F_{ext}=16MHz)
Master mode (see note)	5 MHz * *2.5 MHz* 1.25 MHz * *625 kHz* 312,5 kHz * *156 kHz*	2 MHz 1 MHz 500 kHz 250 kHz 125 kHz 62.5 kHz
Maximum in Slave mode	PIC16C62/72A: 1.78MHz (1/(2.5Tcpu+60ns) PIC16F87x: 2.27MHz (1/(2Tcpu+40ns)	4MHz (Fcpu/2)

*These baud rates use the TIMER2 output as baud rate generator.

• Conclusion

The two SPIs, are almost the same.

The **PIC16 is faster in transmission but slower in reception**. And **TIMER2 is needed** in order to obtain some baud rates.

The PIC16 does not allow the SPI and the I²C to be used simultaneously.

The PIC16 provides **a receive-overflow detection** (reception of a new byte completed before reading of the previous). This can be very useful, and certainly more useful than the write collision detection of the ST72.

The ST72 allows a lower SPI speed, which could be useful with some slow peripherals.

4.8 I²C SERIAL COMMUNICATION

Table 28. I²C characteristic

	PIC16C62B/72A	PIC16F87	ST72254
Address	7-bit/10-bit		
Speed	Standard and Fast mode		
Acknowledge	automatic		
Interrupt	End-of-Byte / Stop detection / Address match		
Master/Slave	**Slave only**	Master/Slave	Master/slave
General call support	**No**	Yes	Yes
Multi-master mode	No	Yes	Yes
Bus arbitration		SW	**Automatic**
Write collision detection	Yes	Yes	Yes
Acknowledge detection	Yes		Yes and **automatic acknowledge failure detection**
Receive overflow detection	**Yes**		No
Bus error detection	No		Yes
Power down activity	Yes (in slave mode only)		Yes in wait mode but not in halt

The PIC16C62B/72A doesn't support the I²C master mode (it can be emulated by software). They also don't support general calls.

4.8.1 Common features:

- 7-bit/10-bit addressing
- End-of-byte transmission flag (with interrupt capability)
- Programmable address
- General call support (ST7 and PIC16F87x)
- Standard and fast mode support (100KHz and 400KHz)
- Automatic acknowledge.
- Detection of transmission in progress (busy flag)
- Start/Stop generation (master mode ST7 PIC16F87x)
- Stop detection with IT (slave mode).
- Interrupt on reception of a good address.

• PIC16

- Multi-master support but bus arbitration must be done by software (PIC16F87x).
- Write collision detection: when a write is done while the previous byte is been shifted (master mode PIC16F87x).
- Start detection with IT (slave mode).
- **Receive overflow detection** (slave mode).
- Acknowledge reception detection.
  In sleep mode the MCU can be woken-up when receiving of a byte or when an address match is completed (in slave mode only).

The PIC16 does not allow the I²C and the SPI to be used simultaneously.

• ST72

- Multi-master support with **bus lost arbitration detection** (ARLO)
- The acknowledge of a general call can be disabled
- The acknowledge sending can be disabled (globally)
- **Bus Error detection**: if a start or stop condition is issued during a byte transfer.
- **Acknowledge failure detection**: reception of something else when waiting for an acknowledge.
- In wait mode, the I²C module continues to work normally (master or slave mode) and the MCU can be woken-up by any interrupt from the I²C module.

4.9 USART/SCI SERIAL COMMUNICATION (PIC16F87X ONLY)

This device is present only on the **PIC16F87x.**

It is able to perform serial communications in the following modes:

- Asynchronous (Full duplex)
- Synchronous – Master (half duplex)
- Synchronous – Slave (half duplex)

- 8/9 bit reception mode
- Framing error detection
- Address detection
- Overrun error detection

Table 29. SCI Baud rate in asynchronous mode

Standard Baud rate	Fosc=20Mhz		Fosc=16Mhz	
	Real Baud rate	%error	Real Baud rate	%error
300	NA		NA	
1200	1221	+1.73	1202	+0.16
2400	2404	+0.16	2404	+0.16
9600	9469	-1.36	9615	+0.16
19200	19530	+1.73	19230	+0.16
76800	78130	+1.73	83330	+8.51
96000	104200	+8.51	NA	
300000	312500	+4.17	NA	

Note: The ST72254 does not have an SCI, but some ST72 devices have one (ex: ST72331J2)

4.10 ISP: IN SITU PROGRAMMING

Table 30. ISP Characteristics

	PIC16C62B/72A	PIC16F87		ST72
Memory	EPROM (OTP / UV)	FLASH		FLASH
ISP	Yes	Yes (two modes)		Yes
Pin needed for ISP	5	4	5	5
Pin used	M_{CLR}, clock, data, V_{SS}, V_{DD}	M_{CLR}, clock, data, V_{SS}	M_{CLR}, clock, data, V_{SS}, PGM	Reset, clock, data, V_{SS}, IP_{SEL}
Entering programming mode	Transition from 0V to +12V on M_{CLR}		Sequence applied to PGM while in reset	Sequence applied to IP_{SEL} while in reset
Supply voltage during programming	**+5V only** (even if the board works at 3V)	Bulk erasing works only at +5V Programming can be done at any voltage in the V_{DD} range	**+5V only** (even if the board works at 3V)	No specific needs, programming and verifying can be done at any voltage in the V_{DD} range
Supply voltage during verify	Verify needs to be done at the V_{DD}min and the V_{DD} max of the application	Not precised in the Microchip documentation	+5V	
Programming method	Program directly loaded into the EPROM (OPT/UV)	Program directly loaded into the FLASH	Program directly loaded into the FLASH	First a program is loaded in RAM, then it is executed, and it takes in charge the Flash programming
Programming Timing	10 ms per 14-bit			15 ms per 16*8-bit
Code protection	Yes			Yes

• Conclusion :

Compared to the ISP of the PIC16C62B/72A (old device), the ST72's ISP is far better because, it does not need a special voltage supply. This point is very important in case of an application at +3V where the MCU must be powered at +5V for programming. For the PIC16F87x it is possible to use the ISP at any voltage in the V_{DD} range, but bulk erasing can only work at +5V.

The **ST72 ISP is also about 6 times faster** (for the same size of code).

Note: Microchip's ISP is presented as a "two-pin ISP" but in fact a third pin is needed to select the programming mode (and this is without counting the M_{CLR}, V_{DD}, and GND pins).

4.11 IN CIRCUIT DEBUGGING (PIC16F87X ONLY)

It is available on the PIC16F87x only.

It uses the in situ programming serial interface (M_{CLR}, V_{DD}, GND, RB7 and RB6).

The debugger functionality takes a part of the MCU resources for its own use:

Table 31. Resources used by the In Circuit Debugging

I/O pins	RB6,RB7
Stack	1 level
Program Memory	Last 100h words
Data Memory	Not precised in the Microchip documentation

No further data is available in the Microchip documentation at the time this document is written.

4.12 RESET PIN

Table 32. Reset

	PIC16	ST72
Direction	Input only	**Bi-directional pin**
Internal Pull-up	no	Yes
Pin filtered	no. But Schmitt trigger input.	Yes
Other	Can receive a +12V to enter ISP mode	(to be taken into account the Reset pin must be pulled low for at least 20 µs)

• Conclusion

The ST72 reset pin is more useful and secure. It allows you to easily put all the board in reset when the MCU generates a Reset (internal or external). Its internal pull-up reduces the number of external components.

4.13 PACKAGE

The SMD version packages are the same for both (SO28).

For the DIP package, the situation is different: Microchip proposes a PDIP28 package and ST a SDIP32 package. Despite the fact that the ST72 package has more pins, it is smaller but cannot be plugged on the standard test board

5 DEVICE SUMMARY

Table 33. Device Summary

		PIC16F87x	ST72254
Architecture		Harvard RISC	Von Neuman CISC
Memory	Program Memory RAM Data EEPROM	Up to 8K*14 (really 8092 instructions) 368*8 accessible through **4 banks** of 128 bytes 128*8 EEPROM accessible only through an indexed mode. **(9 instructions are needed do read one EEPROM Byte)**	8K*8 (-> up to 4096 instructions) 256*8 with no bank switching
Stack		8*13 (cannot be accessed, no push or pop)	128*8
Frequency	External / Internal	20MHz / 5Mhz (F_{OSC}/4)	16 MHz / 8 MHz (F_{OSC}/2)
Instruction	Instruction time: Min / Max Number of instruction Multiplication / Division Number of Addressing Mode	0.2µs / 0.4µs 35 SW / SW 3 (indexed mode need bank switching)	0.250µs / 1.5µs 63 HW / SW 11
Interrupt	Sources / Vectors / Priority Register saved Reaction time Min / Max	13 / 1 PC (The context is not automatically saved) 1.4µs / 2.8µs (This time take into account the code needed to save the context, The difference is mainly due to the number of registers saved)	19 / 7 PC, CCR, A, X 1.25 µs / 2.75µs (the difference between these two times is mainly due to different instruction times)
Voltage range		For Fosc >4MHz : 4.5V-5.5V For Fosc ≤4MHz : 2 V- 5.5 V	3 V - 5.5 V
Oscillator		RC / Ceramic / Crystal	RC / Ceramic / Crystal / internal Clock security system (clk filter, safe oscillator)
Consumption	Typ / Max (Run mode) Power saving mode	10mA/ 20mA @Fext=20MHz one sleep mode	5.5mA / 10mA@Fext=16MHz three power saving modes
I/O	Direction control Mode control Output mode Input mode Power output	Individual Global Push-Pull (except 1 pin true open-drain). Floating / 8 pins may have internal pull-up All outputs are HIGH SINK: V_{OL}=0.6V at 8.5mA	Individual Individual Push-Pull / open-drain (logic level except 2 pins) Floating / Pull-Up with interrupt Normal outputs: V_{OL}=0.5V at 2mA 8 High sink outputs: V_{OL}=0.5V at 8mA

Table 33. Device Summary

		PIC16F87x	ST72254
Watchdog	Time out: Min time / Max time Number of time out values Other	7ms-30ms / 896ms-4.2s (depending of the device) 8 / (the prescaler is shared with TIMER0) Dedicated on-chip RC. The WDT can wake-up the CPU when it is in sleep mode	1.5ms / 98ms (Fosc=16MHz) 64 It is possible to load any value in the timer, to easily choose the time-out period
Timer	Timers PWM/IC/OC usable in the same time Access to the 16-bit register PWM Resolution: Max / at 1KHz/ at 20Khz	2 8-bit timers / 1 16-bit timer 2 PWM (same frequency) or 2 IC (same time base) or 2 OC (same time base) or 1 PWM and 1 IC No simple way to access 16-bit timer **(12 instructions to read the timer**) 10-bit / 1024 / 1000	2 16-bit timers 2 PWM and 2 IC or 1 PWM and 2 OC and 3 IC or 2 One Pulse Mode and 2 IC and 2 OC or 4 IC and 4 OC **security system** (latches of LSB after reading the MSB) 16-bit / 8000 / 200
SPI	Speed: Master / Slave	5 MHz / 4.1MHz Receive overflow detection **The PIC does not allow to use simultaneously the I²C and the SPI**.	2 MHz / 4MHz Writes collision detection:
I²C	Speed / Addressing mode Multi-master / Bus arbitration Other	100 kHz or 400 kHz / 7-bit or 10-bit yes / SW Receive overflow detection **The PIC does not allow to use simultaneously the I²C and the SPI**.	100 kHz or 400 kHz / 7-bit or 10-bit yes / HW bus lost arbitration detection Bus Error detection
USART / SCI	Mode (max speed)	Asynchronous (Full duplex) (max 312Kbit/s) Synchronous (half duplex) (max 5000Kbit/s)	no
ADC	Resolution / Channel / Conversion time Voltage reference	10-bit / 5 / 20µs Internal (V_{DD}) or external Can work in sleep mode, using a On-Chip RC and IT to wake-up	8-bit / 6 / 3µs internal (V_{DD})
LVD	Level Hysteresis	One level for 5V operating Done by holding in reset state for a minimum time	3 levels 250mV of hysteresis
RESET		The reset state is not visible externally	The reset state is visible externally
ISP	Pins needed Way to enter programming mode speed	5 or 6 (depending of the mode used) +12V on Mclr or sequence on PGM 10ms to program 14-bit (one opcode)	5 sequence on IPsel 15ms to program 16*8-bit
ICD	In Circuit Debugging	yes	no

6 WEAK / STRONG POINTS

Table 34. Weak / Strong points

	PIC16F87x	ST72254
Core Execution Speed	**Both MCU have the same execution speed at their maximum frequency (**PIC@20MHz and ST7@16MHz external freq.) - Average assembler benchmark speed ratio: 1.03 (ST7 speed / PIC speed) - C compiler benchmark speed ratio: 1.02	
	↗ **Fast bit manipulation** ↗ **Fast direct memory access** => Fast execution of ***small algorithms***	↗ **Fast calculation (8-bit HW multiplication)** ↗ **Fast and easy indirect memory management** ↗ **Fast and easy constant table manipulation** => Fast and easy implementation of ***complex algorithms***
Power Consumption	↗ **The SPI or I²C are still working in sleep mode (for slave operations)** ↘ Only 1 low power consumption mode ↘↘ To keep a timer active while in sleep, a dedicated oscillator is needed	↗↗ **Lower consumption about 2 times for the same speed** ↗ **4 power saving modes (slow, wait, slow-wait, halt)**
Instruction Set	↗ **Instructions are single cycle (except branch which need 2 cycles)** ↘ Only 3 addressing modes ↘ Manipulation of the only index register is not easy (no direct load of literal value)	↗ **Instructions are slower but they are more powerful (difference between RISC/CISC)** ↗ **Many addressing modes (11) with 2 index registers** ↗ **Fast 8-bit HW multiplication**
Stack	↘↘ No software access ↘↘ Only 8 levels deep	↗↗ **Push/Pop instructions, Stack pointer is R/W** ↗↗ **128 bytes deep** ↘ The Stack pointer cannot be used as an index register **=> Allow recursive calls and heavy imbrication level**
Ram	↗ **More RAM on the PIC16F876 (368 bytes)** ↘↘ Need bank switching (each bank contains up to 96 bytes of user RAM)	↗↗ **No bank switching**
EEPROM	↗ **Allow to store permanent data (256 bytes)** ↗ **It is possible to read or write the program memory** ↘ Difficult read access to EEPROM or FLASH (9 or 12 opcodes needed)	↗ **Direct read access to program memory** ↘↘ No way to store permanent data on the ST72254 but some ST7 have a 256-bytes EEPROM ex: ST72331J2
Interrupt	↘ Only 1 interrupt vector => Software overhead to find the interrupt source ↘ No hardware context saving (need to be done by software)	↗ **Each peripheral has its own interrupt vector** ↗ **7 different vectors => Fast and easy interrupt management** ↘ static priority, but some ST7 have a nested interrupt feature ex.:ST72311R6
Clock	↘ RC oscillator usable up to only 4MHz (Fcpu=1MHz)	↗ **Internal RC (4MHz)** ↗ **External RC oscillator usable up to 14 Mhz (=> Fcpu=7Mhz)** ↗ **Safe clock, Clock filter, with clock spike detection** **=> Allow secure clock management for critical applications**

Table 34. Weak / Strong points

	PIC16F87x	ST72254
Timer	↗ **3 timers: One 16-bit and two 8-bit** ↗ **Better PWM precision (max: 10-bit) for frequencies over 3.9KHz** ↗ **The 16-bit timer can use a dedicated slow oscillator** ↘↘ To keep a timer active in sleep mode, a dedicated oscillator is needed. ↘ No easy way to access the 16-bit timer (12 opcodes to read the timer) ↘↘ Many interactions between the 3 timers: The 2 PWMs must run at the same frequency The prescaler of TIMER0 is shared with the watchdog The output of TIMER2 is needed to generate some baud rates of the SPI	↗ **2 true 16-bit timers** ↗ **No interaction between the 2 timers** ↗ **Better PWM precision (max: 16-bit) for frequencies below 3.9KHz** ↗ **Secure and easy access to the 16-bit timer registers**
I/O	↗ **High sink output for all pins => direct led driving** ↘ Only push-pull output mode available (except for two open-drain pins) ↘ When using the Interrupt-On-Change feature (pins RB4-7) reading PORTB must be avoided	↗ **Push pull or open drain output modes with individual control** ↗ **Floating or internal pull-up input modes with individual control** ↘ Activation of the internal Pull-up, active also activates the interrupt. ↘ Only 8 high sink I/Os
SPI	↗ **Reception overrun detection** ↗ **Transmission speed up to 5MHz (@Fosc=20MHz)** ↘ Reception speed only up to 2.27MHz (@Fosc=20MHz) ↘ Need TIMER2 output for some baud rate ↘↘ Can not be used at the same time as the I²C cell	↗ **Reception speed up to 4MHz (@Fosc=16MHz)** ↘ Transmission speed only up to 2Mhz (@Fosc=16MHz)
I²C	↗ **Reception overrun detection** ↘↘ Can not be used at the same time as the SPI cell	↗ **Multi-master support with hardware bus arbitration**
USART / SCI	↗ **Present**	↘↘ No SCI on the ST72254. But some ST7 have a SCI cell ex: ST72331J2
ADC	↗↗ **10-bit Resolution** ↗ **Automatic sampling rate with low software overhead** ↘ Slow conversion (20µs, up to 50KHz)	↗ **Fast conversion (3µs, up to 333KHz)**
In System Programming	↘↘ Fully functional at Vdd=+5V ONLY	↗ **Faster (about 6 times)** ↗ **Works at any V_{DD}level**
In Circuit debugging	↗↗ **Present (but not yet fully described in the Microchip documentation)**	↘ No
Voltage Range	↗ **Larger voltage range (2V-5.5V)** ↘ 2 different devices in order to cover the entire voltage range ↘↘ Below 4.5V, the external frequency must be decreased to 4MHz	↗ **between 3.5V and 5.5V the maximum frequency can be used (16MHz)** ↘ smaller voltage range (3V-5.5V)

"THE PRESENT NOTE WHICH IS FOR GUIDANCE ONLY AIMS AT PROVIDING CUSTOMERS WITH INFORMATION REGARDING THEIR PRODUCTS IN ORDER FOR THEM TO SAVE TIME. AS A RESULT, STMICROELECTRONICS SHALL NOT BE HELD LIABLE FOR ANY DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WITH RESPECT TO ANY CLAIMS ARISING FROM THE CONTENT OF SUCH A NOTE AND/OR THE USE MADE BY CUSTOMERS OF THE INFORMATION CONTAINED HEREIN IN CONNEXION WITH THEIR PRODUCTS."