Don Lancaster's

RESOURCE BIN

number fifty four

Picking a potent peck of pliant PIC's.

Our usual reminder here that the *Resource Bin* is now a two-way column. You can get tech help, consultant referrals and off-the-wall networking on nearly any electronic, *tinaja questing*, personal publishing, money machine, or computer topic by calling me at (520) 428-4073 weekdays 8-5 Mountain Standard Time.

I'm now in the process of setting up my new *Guru's Lair* web site you will find at (where else?) *www.tinaja.com* This is the place you go for instant tech answers. Among the many files in our library, you will find complete reprint sets for all of the *Resource Bin* and other columns. Plus a brand new Synergetics Consultant's Newtwork & lots of links to unique web sites.

You will get the best results if you have both *Netscape Gold* and *Acrobat Reader 3.0* installed. This new reader does utterly amazing things online.

Meet the PIC

This month, I thought we'd see just what the excitement is all about. The PIC microcontroller appears to have become *the* chip of the decade. To the point where there's no point *ever again* designing *any* bits-and-pieces classic hardware solution.

Especially one which involves 555 timers or other jellybean chips.

A PIC is basically a tiny black box that has lots of digital (and sometimes analog) inputs and outputs. You teach the box what relationship you want to have between your input and outputs. This gets done by writing a program or a command series on a PC host.

Those commands then get sent to a programmer which blasts instructions into one or more PIC chips. Your PIC then remembers what it is supposed to do. It can then be used anywhere.

There's several big time advantages to PIC's that utterly and totally blow their competition away. It is not even remotely close.

First and foremost, PIC's are readily available in smaller quantities. And totally hassle free. To individuals, to students, to small scale tech startups. Anybody. While typical onesie-twosie costs are $4 to $7, the prices on some chips drop under a dollar.

Second, a PIC uses a non-traditional *Princeton* RISC architecture. In which your program and data memories are (usually) held strictly separate. These longer self-addressing instructions let you do most computer commands as *one byte* instructions which execute in *one single clock cycle!*

This one-on-one feature gives you as much as a 3X speed advantage and up to a 3X code length advantage over classic 68XX or 80XX micros.

Third, the PIC is elegantly simple. There are scant few commands in the PIC instruction set. But they are super powerful and very easy to use.

Fourth, your PIC can be quite fast. Because of one byte commands. And because of 20 MHz clocks.

Fifth, the PIC is a very low power device that is easily battery operated. With careful clock design, it can even become the core of your micropower system run off coin cells.

Finally, the PIC is extremely easy to learn and is lots of fun to use.

NEXT MONTH: Don shows us a few insider secrets on net webmastering.

Additional specific technical details on what a PIC does and how it does it are in MUSE88.PDF and MUSE99.PDF on my new *www.tinaja.com*

Let's try to interleave the PIC basics with today's major resources...

How Fast is Fast?

When you write custom crafted and hand-coded machine language, a PIC chip might end up astonishingly fast. Sadly, most designers go out of their way to *slow their PICs down*.

Assume you want to quickly shove a square wave out a port. Start with a 20 MHz PIC. By using internal machine language only, we toggle the port and then continuously repeat.

There are four PIC clock cycles per machine cycle. Thus, a 20 MegaHertz clocked PIC executes each instruction in a blazing 200 nanoseconds.

Allow one machine cycle for your toggling, and two for the goback. Our square wave has a half period of three cycles or a full period of six. The port output frequency is 833 kHz.

More realistically, you'll want to do a decrement or test of some sort that gobbles up at least one extra cycle and lowers you to 625 kHz.

That is *if* your code gets internally stored. Your normal way to expand a PIC's memory storage is with serial EEPROM. This route is cheap, simple, and needs few interconnects.

But dozens of clock cycles may be involved. Hundreds when you've got to write to memory as well.

If you are using external EEPROM memory, your square wave's output frequency will drop on down into the 50 kHz range.

Most any computer program can be *interpreted* or *compiled*.

With any interpreted program, each program step gets read in order and then converted into a useful action *at run time*. This interpreted program is often first written in a chosen higher level language as *Basic* or *PostScript*. These are easy to write and interact with humans well.

With a compiled program, an initial *compiling* run is done to sort out only your essentials of exactly what has to be done. Only the precise instructions needed at run time get used.

The rule is that machine compiled code is usually faster than interpreted

code. And that hand-crafted machine language is way faster than either.

Continuing the speed examples, the *Basic Stamp* from *Parallax* is still by far the most popular PIC interpreter. An original stamp could output a square wave at 500 Hertz or so.

The key point is that interpreted vs hand generated speeds are not even remotely close. They differ by a factor of 10 to 100 or even more.

Always hand code for speed!

A PIC running at the usual 8 MHz slurps around ten mils of its supply current when active. This may be far too much if you are trying to run off a tiny coin cell.

The PIC current is proportional to clock frequency. Typical micropower apps often substitute a 32 kHz clock. Thus trading speed for power.

Your usual route towards a fast PIC design is to do a slow design first. Using your Basic Stamp.

This step makes sure your system does what it is intended. When only a modest further speedup is required, you substitute Scott Edwards or other compiled routines for each step in the program. For dramatic speedups, you rewrite the whole thing from the ground up in custom crafted machine language. Like you really should have done in the first place.

Microchip Technology

These people are the prime supplier for PIC chips and tools.

They have got dozens of data books and ap notes available. Start off with their free *PIC Data Book* and the fine *Microcontroller Applications Manual.*

Originally, there were three main PIC families. The 16C54 (and the other 16C5X chips) were their cheapest and simplest. They typically offer 12 to 20 port lines, 512 to 2048 bytes of fixed program memory and 25 to 73 bytes of user RAM. With 12 bit opcodes.

Don't let all these seemingly small values fool you. PIC code takes up far less memory than is needed for classic micros. And you can always easily tack on great heaping megabytes of external serial EEPROM.

Their 16CXX series has additional memory. Program memory to 4096 bytes and up to 192 bytes of RAM. The instruction words are 14 bytes wide. They also do include a full time serial port to simplify programming.

The high end 17CXX series offers a full 64K of memory space, interrupts, hardware multipliers, and all the rest of the bells and whistles.

And a brand new series of ultra low cost PIC's has just been announced in *eight pin* packages for seriously low end apps. Priced well under a dollar in humongous quantities.

Microchip Technology also has tech support, plenty of software, in-circuit emulators, and design seminars.

Parallax

Lance Wally and his *Parallax* folks are the home of the *Basic Stamps.* Tiny modules that make PIC's friendly and ultra easy to personally use.

Their original *Basic Stamp I* was an *entire* microcontroller in a $29 14-pin SIP package. Measuring half an inch high by an inch and a half wide.

This BS1 gave you eight I/O lines, 256 bytes of program memory, and ran 2000 instructions per second.

Their newer BS2 is in the shape of a 24 pin integrated circuit. It has 16 I/O lines, executes 4000 instructions per second, and handles serial data to 50 kilobaud. It also has new commands that produce DTMF tones and the X10 remote control instructions.

Parallax has also got lots of support products. From hardware interfaces to *Stamp Experimenter Boards* and full programmers. For serious designers, they also offer a series of *ClearView* in-circuit emulators.

That Parallax web site can be found at *www.parallaxinc.com* Do note that *inc* in the middle of their url address! *parallax.com* is somebody else. At any rate, they've got the full *Basic Stamp* ap notes, programming software, and bunches of other goodies available for your immediate downloading.

There's also a hot link to here from my *www.tinaja.com*

Scott Edwards Electronics

Scott Edwards is another *Nuts and Volts* author. You will find his Basic Stamp column and PIC Applications info right here on a monthly basis.

Scott is heavy into *PIC Development Tools*, his series of machine language software modules which do BS things much faster and far more compactly. They also extend the Stamp's memory and I/O capabilities

Scott also carries various *backpack* add-on hardware goodies. As liquid crystal displays, thermometers, and applications that involve the low cost radio control servos.

Scott also does custom consulting and application development.

I've got reprints of many of Scott's columns up on my *www.tinaja.com*

SOME PIC RESOURCES

Adobe Acrobat
PO Box 7900
Mountain View CA 94039
(800) 833-6687

E D Technical Pubs
PO Box 541222
Merritt Island FL 32954
(407) 454-9905

Microchip Technology
2355 W Chandler Blvd
Chandler AZ 85224
(602) 786-7200

Nuts & Volts
430 Princeland Ct
Corona CA 91719
(714) 371-8497

Science First
95 Botsford Place
Buffalo NY 14216
(800) 875-3214

Axiom Manufacturing
717 Lingco Drive #202
Richardson TX 75081
(214) 994-9676

Electronics Now
500-B Bi-County Blvd
Farmingdale NY 11735
(516) 293-3000

microEngineering Labs
Box 7532
Colorado Springs CO 80933
(719) 520-5323

Parallax
3805 Atherton Rd, #102
Rocklin CA 95765
(916) 624-8333

Synergetics
Box 809
Thatcher, AZ 85552
(520) 428-4073

CCS
PO Box 11191
Milwauke WI 53211
(414) 781-2794

Scott Edwards
964 Cactus Wren Lane
Sierra Vista AZ 85635
(602) 459-4802

Midnight Engineering
1700 Washington Ave
Rocky Ford CO 81067
(719) 254-4558

Ken Pergola
2088 Swamp Road
Richmond MA 01254
(413) 698-3167

Advanced Transdata
14330 Midway Road #128
Dallas TX 75244
(214) 980-2960

Circuit Cellar Ink
4 Park Street #20
Vernon CT 06066
(203) 875-2751

ITU Technologies
3477 Westport Ct
Cincinnati OH 45248
(513) 574-7523

Netscape Gold
501 E Middlefield Rd
Mountain View CA 94043
(415) 528-3777

Picard Industries
Box 61
Clarendon NY 14429
(716) 589-0419

Zeta Electronic Design
18 Bismark Street
Manchester NH 03102
(603) 644-3239

Advanced Transdata

Any PIC device is a progammable microcontroller. You have to teach it what it is going to do. You have four main choices here: You can use a OTP or *one time programmable* chip. These are the cheapest *for working and tested code in small to medium quantities.*

The earliest PIC's required a fairly specialized programmer driven from a host's parallel port. Later versions use their serial I/O to greatly simplify programming and host interface.

When you are certain you will sell a minimum of 25,000 identical copies of your chip, factory programmed ROM versions are available.

What if you intend to develop or improve your code? There are more expensive EEPROM versions of PIC's available. You could reprogram these over and over again. But when you are really serious about doing major PIC development work, you'll want to go to an ICE or *in-circuit emulator*.

An emulator fakes PIC behavior to the best of its ability. The simplest of emulators is more properly called a *simulator*. Simulators execute only a software model on a host computer, usually on a PC. Very slowly.

Instead, true *emulators* hide a real PIC somewhere in their innards. A mid-priced emulator might run quite slow. Perhaps at only 1/100th of its normal speed. While including such niceties as single stepping, tracing, debugging, and monitoring of key variables and data locations.

A *real time emulator* does all this and does it at your normal and expected PIC speeds. Finaly, an ICE, or *in circuit emulator* has a physical plug on it you drop into your PIC socket.

The real time emulators are usually based upon the special fancy PIC chip known as a *bond out version*. In which additional access points are brought out to multiple pins. The real beauty of emulators and simulators is instant and continuous reprogramming.

The fancier in-circuit emulators are not cheap. Pricing in the $600 to $1000 range is typical. Put another way, an in-circuit emulator can easily pay for itself in a few dozen hours of avoided design time.

While Microchip and Parallax both sell great in-circuit emulators, lately I've been using the *RICE16-5X* from *Advanced TransData*.

Advanced Transdata carries a nice collection of higher end development tools. Mostly centering on emulators and programmers. Plus their RICE16 development software.

Along with several other suppliers, they also offer C compilers. But note that *any* code you compile from the C language is *guaranteed* to end up as an *absolute atrocity* when it is run on any PIC system! Every time.

The reason is simple: Unless you know, fully grok, and genuinely love both PIC machine language code and its architecture, you're *certain* to get lousy results. That C language goes out of its way to lock you out of this essential understanding process.

microEngineering Labs

This is Jeff Schmoyer's operation up in Colorado Springs. Jeff is a superb source for the PIC's themselves and for low cost PIC prototyping boards. A 16C54 PIC sells for $4.50 in singles, while their *PIC Proto-18* breadboard goes for $9.95.

Jeff is also big on *PBasic Compilers*. Host software which precompiles the PIC commands so they will run much faster than interpreted Basic. $99.

Among dozens of other products, Jeff also has a $29 *PIC Proto Demo Kit* which includes a PIC, a breadboard, LED's, a speaker, and a collection of related goodies.

Circuit Cellar

The Steve Ciarcia *Circuit Cellar* has long been a quality source of low cost microcontroller systems. Especially those suited for home automation.

Steve's latest product is called the *Pic Stic1*. This combines their own hardware with software and tools by microEngineering Labs.

There are three devices offered. The first one is BS1 pin-for-pin compatible. Being interpreted code, this beats the BS1 by up to fifteen times on speed. The second one adds a real time clock, while the third adds some fancy A/D conversion capabilities.

These are not yet BS2 compatible. They are working on newer versions that shortly will be.

Science First

These folks are newcomers to the PIC arena. Their main gig is selling low cost Van DeGraff generators and educational science demo projects to students and teachers.

They've just introduced a very easy to use PIC trainer and some simple real-world interface stuff. Intended mainly for non-programmers. Typical are relay drivers, displays, burgular alarms, EPROM emulators, and a few others. Prices from $20 to $60.

Contact Nancy Bell at *Science First* for free catalogs and newsletters.

Synergetics

Uh, thats me, I guess. I'm strongly attracted to the PIC chip because of it being the heir apparent to the 6502, its elegant simplicity, and its ultra clean code. I've been mainly developing in two areas: Flutterwumpers and magic sinewaves.

As we've seen in previous columns, a flutterwumper can be anything that moves and either chomps or spits under intelligent control. X-Y tables, circuit drilling, engraving machines, signmakers, Santa Claus machines, or CAD/CAM mills.

The fine PostScript general purpose computer language is absolutely ideal for a flutterwumper's high level side. Among zillions of other reasons, there is PostScript's incredible font variety and its graceful curves.

To me, it makes the most sense to split the problem. Use PostScript on a suitable host to initially generate your flutterwumper paths. Convert those paths to some simple meta language. Then let a PIC or two down on the flutterwumper do all of the low level scut work for you.

While you can use HPGL or Gerber format, to keep costs really down, all you'd really need is a simple code of single ASCII characters. 0 for north, 1 for northeast, and so on. U for up and D for down. Q for quit.

By splitting PostScript on host and PIC on flutterwumper, a $100 printed circuit drill and a $200 Santa Claus machine should be possible.

Magic sinewaves are some newly discovered and ultra-long repeating sequences of ones and zeros. They have precisely controlled amplitudes and amazingly low distortions.

Major new uses include induction motor speed controls, electric cars, solar panels, power inverters, and home energy efficiency improvers. Compared to PWM, magic sinewaves offer lower costs, higher efficiencies, cooler operation, much less rfi, and elegantly simple designs.

I will be happy to send you a free reprint on my magic sinewaves if you call, email, or write. Or a full blown development proposal if you have more than casual interest. Seminars, sourcecode, and my full consulting services are available.

Additional files on flutterwumpers, Santa Claus machines, and my magic sinewaves on *www.tinaja.com*

Also Rans

The PIC is absolutely ideal for small scale tech startup ventures. Let's look at a few random samples…

Axiom Manufacturing offers a *PIC Microprocessor Development Kit*. *CCS* has a group of *PIC* related products, both programming and development oriented. Fred Eady's *E. D. Technical Publications* specializes in low cost PIC development tools.

ITU Technologies sells the *PIC-1 Programmer*. The kit is $29 while the assembled and tested version goes for $49. They also do retail individual PIC chips and are heavy into caller id and similar PIC applications.

Matrix Labs is the home for the *QuicPic Language* and their new *16C57 Proto Boards*. Ken Pergola is the source for his *Micro-bRISC* programmers, software, and useful firmware. *Picard Industries* sells a number of interesting and low cost *PIC* development aides.

As does *Zeta Electronic Design*. And a few others on the PIC library shelf of my *www.tinaja.com*.

This Month's Contest

As our contest for this month, just tell me about any PIC resource I don't already know about. Or suggest some new, unique, or totally off-the-wall application for a PIC design.

There will be a largish pile of my new *Incredible Secret Money Machine II* books going to the dozen or so better entries, plus an all-expense-paid (FOB Thatcher, AZ) *tinaja quest* for two that will go to the very best of all.

Send all your *written* entries to me here at *Synergetics*, rather than to *Nuts & Volts* editorial. ✦

Microcomputer pioneer and guru Don Lancaster is the author of 33 books and countless tech articles. Don maintains his no-charge US tech helpline found at (520) 428-4073, besides offering all of his own books, reprints, and consulting services. Don also has two free catalogs full of his resource secrets waiting for you. Your best calling times are 8-5 on weekdays, Mountain Standard Time.

Funding and time constraints restrict this helpline service to US callers only.

Don is in the process of setting up his Guru's Lair at http://www.tinaja.com

Full reprints and preprints of all Don's columns and ongoing tech support appear here. You can reach Don at Synergetics, Box 809, Thatcher, AZ 85552. Or send any messages to his US Internet address of don@tinaja.com