Don Lancaster's

Tech Musings

March, 1996

PIC programming tricks
Greeting card music chips
Update for Adobe Acrobat
Selecting addressing modes
Online TV engineering sites

There sure has been quite a lot of new microcontroller interest both on PSRT and my tech helpline lately. This month, I figgured we might take a look at the secrets to understanding and using new low end micros.

This can involve the learning and mastering of microcontroller…

Addressing Modes

Each company has different names for them, and each places different emphasis on how important each one is. Regardless, a computer *address mode* is some way the CPU central processing unit has of reaching data or program memory locations both within and outside its *address space*.

As we have seen in some previous columns, the address space represents the total reach of the microprocessor. Those individual byte locations in an address space typically support ROM (which is memory that is more or less permanent); RAM (memory which is fast and easy to change); I/O (short for input/output); or nothing at all (unused). More on address space in MUSE92.PDF.

To understand a micro, you *must* understand its underlying addressing modes. Use the wrong mode and you are *certain* to get into trouble. And not picking the "best" mode for your task may cause your program to need more memory. Or to run slower.

As I see it, there are seven basic addressing modes that are in popular use today. I have summarized these in figure one. They are…

implied– With implied addressing, no further information is needed to complete the task. Such as a CLC that clears a carry flag. Or does a similar housekeeping duty. Say you just got home, and there, sharpening its claws on your favorite speaker grill, is the cat. Exactly what you say probably won't be printable. But the chances are it will leave no doubt whatsoever *which* cat and *which* speaker grill you are referring to.

Implied addressing can quickly do specific functions, but its powers are otherwise limited. These are usually single-byte instructions.

immediate– By using immediate addressing, a fixed *value* gets placed into an accumulator, a register, or in another location. Such fixed values are sometimes called *literals*. Your fixed value either gets built into the instruction opcode word (PIC) or can be the byte immediately following an opcode (everybody else).

You can think of a gas station as immediate addressing. Where you place exactly an ordered quantity of gas into your vehicle.

Immediate addressing is the way of introducing fixed values into your programs. But when you need one of many possible values, a fancier mode will be required.

relative– With relative addressing, you'll jump so many steps forward or backward from *where you now are*. Score usually gets kept by a *program counter*. Normally, the PC advances just one address location at a time, picking up as many bytes as may be

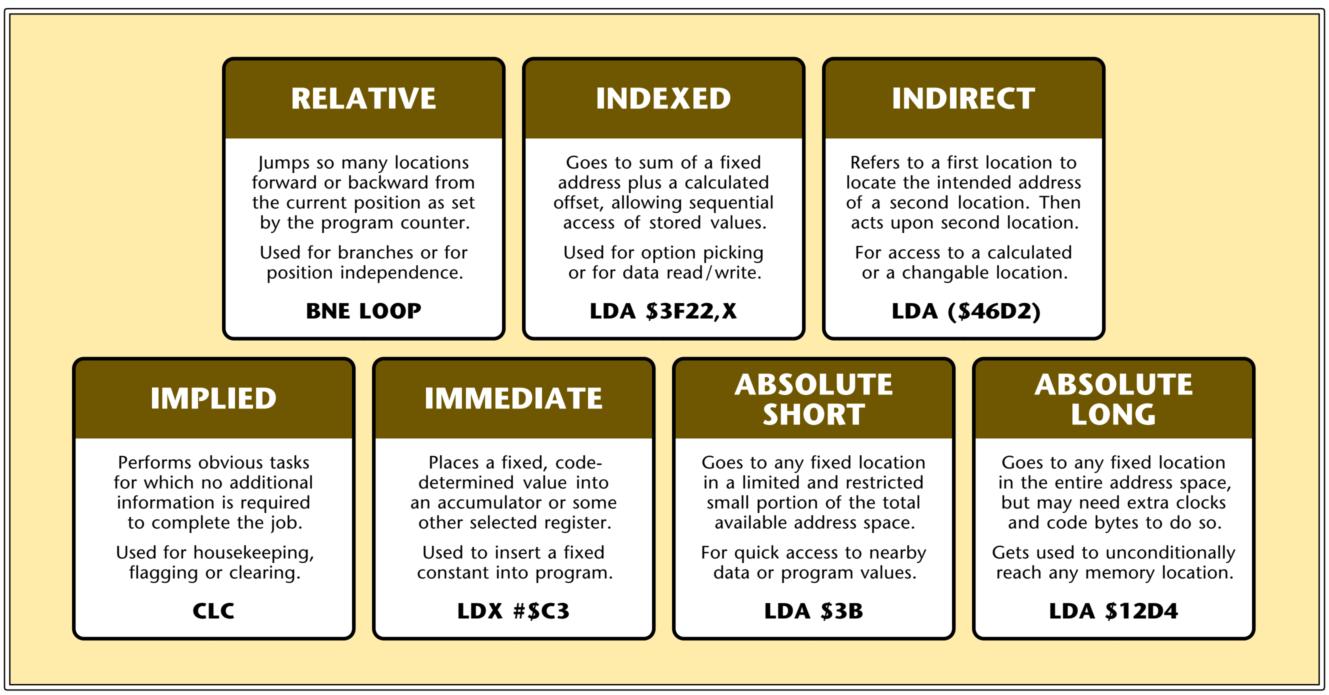

Fig. 1 – THE KEY SECRETS to using and understanding microcontrollers and microprocessors lie in understanding and properly selecting addressing modes. Here are the seven most common addressing modes in use today.

INDEXED–

The PIC lets you modify the bottom 8 bits of the program counter, just like it was any other register. Here's how to select one of four options…

```
PICKONE  ADDWF PC,1   ;add selection 0-3 to PC
         GOTO  SEL0   ;selection 0 if chosen
         GOTO  SEL1   ;selection 1 if chosen
         GOTO  SEL2   ;selection 2 if chosen
         GOTO  SEL3   ;selection 3 if chosen
```

INDIRECT–

Indirect addressing works the same way, by modifying the program counter. Here's how to go to a calculated **NEWADD** address…

```
GOTOADD  MOVFW NEWADD ;calc address to W
         MOVWF PC     ;change program counter
                      ;continue at NEWADD.
```

RELATIVE–

It is unlikely you would want to go to a relative address, because it is just not a PIC thing to do. But, if you must…

```
BRANCH   MOVFW HOWFAR ;branch offset to W
         ADDWF PC,1   ;send PC there
                      ;continue at branch
```

Flag-specific opcodes are thankfully absent in the PIC. Instead, the superpower test-any-bit-and-skips do the job. Here's how to branch on carry clear…

```
TESTC    BTFSC S,C    ;test carry flag
         GOTO  CSET   ;move here if set
         GOTO  CCLR   ;move here if clear
```

It is often possible to avoid relative addressing entirely. By using an "if the left one don't get ya, the right one will" ploy. Such as this constant-time example to move any random bit into any arbitrary location…

```
MOVB     BTFSC AA,BB  ;test bit BB of register AA
         BSF   YY,ZZ  ;set bit ZZ of reg YY if 1
         BTFSS AA,BB  ;test bit BB of register AA
         BCF   YY,ZZ  ;set bit ZZ of reg YY if 0
```

PROGRAM DATA ACCESS–

The PIC's "non-Von" architecture separates program and data memory. But the RETLW command can return literal data stashed on the program memory side. Here's how to convert integers 0-15 to ASCII characters 0-F: Start by building up this table lookup data stash…

```
STASH    ADDWF PC,1   ;go to selected char
         RETLW #$30   ;ASCII 0 if hex 0
         RETLW #$31   ;ASCII 1 if hex 1
            . . .         . . . . .
         RETLW #$45   ;ASCII E if hex E
         RETLW #$46   ;ASCII F if hex F
```

When you need it, use this sub to convert hex in a mere five machine cycles. Begin with 0-15 in the W register…

```
         CALL STASH   ;returns ASCII to W
```

Fig. 2 – THESE "MISSING" ADDRESS MODES are easily added to any PIC.

required for any instruction and then going on to the next one. Exceptions occur whenever you jump to a chosen new location or go to a subroutine or accept an interrupt.

Relative addressing is commonly associated with *branches* and *loops*. Often, some test is made, such as a BNE branch-if-not-equal. On branch not taken, the program counter goes on to the next instruction. On branch taken, a relative hop gets made.

Relative addressing is the same as using a pogo stick. Go so many hops forward or backward from *where you now are*. One interesting property of relative addressing is that it is usually *position independent*.

absolute long– With absolute long addressing, you can reach anywhere in the entire address space. Just like a town zoning map, any house can be located within your city limits. This mode typically takes several bytes. In a 64K address space, you might need one byte for the opcode, one byte for the lower 8-bits of your address and a final byte for the upper 8-bits.

Thus, absolute long addressing can end up slow, long, and ungainly. It is also usually limited to going to one *fixed* and *known* location.

absolute short– With an absolute short addressing, you can quickly and conveniently get to some *assumed smaller portion* of the address space. Such as your lowest 256 locations in memory. Or the current memory page you were just working on. Say you are hanging a painting on your living room wall. You usually don't worry about *which* house on *which* block is involved. You'll just assume you are already in the right house.

Absolute short is normally faster than absolute long and often can need fewer program bytes. But its reach is obviously limited. And turf fights can place high value on certain locations.

indexed– Things do get interesting should you want to rapidly access a bunch of *nearby* memory locations. Like going into the donut store and ordering one of each. The clerk will take one from each tray as they go along. With indexed addressing, you reach the *sum* of a *fixed* base address and a *variable* offset. Reading a file message is one obvious use.

Indexing could get done by adding the contents of an *index register* to a base address in the operand. Or can be done by direct modification of the program counter.

indirect– Sometimes you'll want to *calculate* an address location. To do this, you use an indirect address. Where you first *refer* to one address to get a second one. Just like asking the little old lady in some rural Post Office where somebody lives.

Indirect addressing often gets used for *option picking* or *case* commands, where a menu selection can get you to several different locations. This is also useful for larger tables where the *start* of a message or a data sequence has to be predetermined. Depending on the micro, indexing can be 8-bits or 16-bits wide.

Assembler Rules

Most often, you'll create all your machine language code by using an *editor* and an *assembler*.

Each assembler will have a certain rule set that lets you pick an address mode. The rules are often based upon the number of hex bytes involved and certain punctuation symbols in your operands. One of the possible sets of rule has been used in the assembler notation examples of figure one.

*Watch all assembler operands **very** carefully!*

For instance, a hex location will either start with a "$" or end with a "H". A "#" usually selects a literal value. Absolute long addresses might be four hex bytes long. Instead, an absolute short might only be two hex bytes. Indexing is sometimes shown with a comma, while indirect often involves (parenthesis).

Needless to say, it gets extremely important to know and understand the rules for the particular assembler you are working. The most common beginner mistake is mixing up hex and decimal location values.

Locations $09 and $10 are *not* beside each other! Another goof is to leave your "#" off a literal, changing you over to absolute short addressing instead. In one case, you get value #$06. In the other, you get whatever location $06 may happen to feel like giving you at this particular time.

With wildly different results.

PIC Addressing

Sometimes it takes a little extra time and effort to master the address modes which are *really* available on any micro. Two good ways to do this are to study existing code and to try out new things on your own.

We saw an example of how to pick up indirect indexed addressing on a 6805 a few columns back. Details in MUSE94.PDF on *www.tinaja.com*.

At first glance, those stupendously great PIC microcomputers seem to lack much in the way of addressing modes. Especially the low end chips. In fact, all you'll initially see is one plain old absolute short, fixed code GOTO command.

```
; Code module to multiply a BASE integer 0-21 by a factor
; of 12. Can be used for the access of tablular data.

X12        BCF S,C          ; clear carry flag
           RLF BASE,1       ; 2X BASE into BASE
           RLF BASE,1       ; 4X BASE into BASE
           RLF BASE,0       ; 8X BASE into W
           ADDWF BASE,1     ; 8X+4X = 12X into BASE
```

Fig. 3 – PIC SOURCECODE can be amazingly short and fast. Such as this five-byte and five-cycle multiply by twelve….

What tricks could you play here? Figure two summarizes some sneaky ways to astoundingly multiply the flexibility of PIC addressing.

The absolute short addressing on the 16C5X accesses 512 bytes for a GOTO or 256 bytes for a CALL. Your other higher 512-byte memory pages can be reached by setting or clearing page bits in the status register.

Or, on fancier PIC's, modifying a register known as PCLATH gives you similar absolute long addressing.

A unique feature of the PIC family is that the lower 8 bits of the program counter might be written to just like writing any other register! As figure two shows us, direct PC access gives you a method to do relative, indexed, and even indirect addressing. As well as accessing program-side data.

For instance, when you store some value to your PC program counter, it acts just like a GOTO. If, instead, you *add* something to PC, you can move so many steps forward (or backward) for indexing or branching.

One curious gotcha, though: On low end 16C5X devices, either a sub CALL or a program counter PC write *forces* address line A8 to *zero*. This means you'll have to *carefully* plan what goes where. And that you are limited to only *one-half* of your total available memory for data or for PC based moves. But note that you can always "Get out of Dodge" on a plain old sub. Just call a GOTO.

The fancier PIC chips do not have these restrictions.

Two Examples

The PIC is rapidly becoming *the* chip of the decade. No matter what your electronic project, you can make it better, faster, and cheaper if you add one or more PIC chips to it.

Those PIC's *completely* blow the competition away. Through 3X speed and 3X code length advantages. But even more by its ultra-clean elegant simplicity. Nearly all instructions are one single byte long and execute in one single machine cycle.

Let us look at an example or two that might give you the flavor of PIC programming. We've already looked at a *six* byte and *six* clock sinewave

```
; Code module to provide a time delay of n instruction
; cycles. n can vary from 4 to 255 and is destructively
; read from COUNTER. There are 2 or 3 cycles of overhead.

; The module is easily extended to 9 or 10 bits.

DELAY     BCF S,C            ; clear carry (if needed)
          RRF COUNTER,1      ; bit 0 into carry
          BTFSC S,C          ; stall one cycle?
          BCF S,C            ; yes, but MUST clear carry
          RRF COUNTER,1      ; bit 1 into carry
          BTFSC S,C          ; stall two cycles?
          GOTO LOOP          ; yes, by using extra GOTO

LOOP      BCF S,C            ; 4*n cycle delay loop
          DECFSZ COUNTER     ; round and round
          GOTO LOOP          ; till done
```

Fig. 4 – …or perhaps this TIME DELAY GENERATOR that offers a resolution of ONE single instruction cycle.

ONLINE TELEVISION ENGINEERING RESOURCES

MANUFACTURERS–

http://www.hp.com	Hewlett-Packard
http://www.sony.com	Sony
http://www.tektronix.com	Tektronix

NEWS GROUPS–

sci.engr.television.broadcast
sci.engr.television.advanced
rec.arts.tv.interactive
rec video.satellite.dbs
rec video.satellite.europe
rec.video.satellite.misc
rec.video.satellite.tvro

ORGANIZATIONS–

http://www.sbe.org	Society of Broadcast Engineers
http://www.smpte.org	Society of Motion Picture & TV Engineers

generator in MUSE85.PDF.

In my magic sinewaves, I needed a multiply by twelve. To calculate the table base addresses. As figure three shows us, our PIC does this in five bytes and five cycles. Four and four actually. If you have a known clear carry flag on entry.

Note that most PIC instructions are *dual mode*. An assembly command ending in ",0" places the result back into your "W" *accumulator*. But one ending in ",1" places the result into the target register instead.

One urgent need common to many microcontroller programs is a high resolution delay generator. These are handy for everything from electronic music up to baud rate generation to timing circuits. Your obvious starting point with any micro is a countdown *loop*. The trouble is that any loop will take several machine cycles for its decrement-and-branch.

Can we do better?

Your key secret appears in figure four. Purposely slow down your loop so it takes exctly *four* cycles. Before using your loop, first shift the LSB of the count value into the carry. Then add *one* cycle of delay if carry is set. Next, shift the next count value bit into the carry, and selectively add *two* cycles of delay if the carry is set again. Finally, use the main loop for multiples of four cycles.

Presto. A delay generator having a one cycle resolution! Done in nine or ten bytes of code. With a mere two or three cycles of overhead.

Uh, that strange BCF S,C in the loop serves two purposes. Firstoff, it acts as a NOP to burn up one cycle of delay. And second, it acts as a "good neighbor" to leave a cleared carry for later use elsewhere.

That first BCF S,C clearing can be omitted if it is never needed.

One free *Incredible Secret Money Machine* if you can show me *any* way to shorten or otherwise improve this code. On *any* micro.

To get yourself PIC literate, start with that *Microchip Data Book* and the new *Microcontroller Applications Manual* from *Microchip Technology*. Then go to the *Basic Stamp* products from *Parallax*. Finally, step up to the *Scott Edwards PIC Tools*.

I've got scads of PIC support up on my *www.tinaja.com*. Including a lot of the *Scott Edwards* tech info.

NEED HELP?

Phone or write all your US Tech Musings questions to:

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

US email: *don@tinaja.com*
Web page: *www.tinaja.com*

Online Television Engineering Resources

TV Technology is a great trade mag from those same people who publish *Radio World* and *Computer Video*. In a recent issue, they included a handy rundown of the major internet sites of interest to television engineers.

I've excerpted part of their listing as this month's resource sidebar.

Acrobat Update

The folks at *Adobe Systems* have dramatically improved and upgraded the fine device-independent *Acrobat* document distribution system. They even provide a free new *Acrobat CD Sampler* available on request. Well, sometimes it is free and sometimes there is a modest shipping charge. Their sampler includes free Acrobat readers for Mac, Windows, DOS and UNIX. Plus lots of other goodies.

The key to Acrobat is a more or less ordinary textfile called a .PDF file. This file format preserves the *exact* layout and typography of any document. Technical or otherwise. Compared to grubby old fax, you get *camera-ready* copy much faster and cheaper. Compared to HTML, you get infinite more flexibility and *exact* control of what your user sees.

Anyone can print or view a .PDF file by booting one of the free and widely available readers. The .PDF files can be generated by any major application that uses *PDF Writer*.

You can make your own files from any PostScript input by using the *Acrobat Distiller*. You modify your files for distribution using *Acrobat Exchange*. More in MUSE94.PDF, in ACROCAT.PDF, and in my reprints.

Let's see. The neat new stuff. One is a *Weblink* plug-in. Making Acrobat online compatible with *Netscape* and other surfing software. You can now *directly* view all downloaded Acrobat documents while online. Their ability to navigate through any document is also greatly improved.

There's now a new *article* feature made up of *threads* and *beads*. These literally let you string together bits and pieces of text.

And a new *movie* feature lets you do animation, *Quicktime* movies, or even quality sound bytes.

NAMES AND NUMBERS

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

Dicker International
225 Broadway Ste 2000
New York NY 10007
(212) 962-3232

Scott Edwards Electronics
964 Cactus Wren Lane
Sierra Vista AZ 85635
(520) 459-4802

GEnie
401 N Washington St
Rockville MD 20850
(800) 638-9636

Indium Corp of America
1676 Lincoln Avenue
Utica NY 13502
(315) 768-6400

Industrial Market Place
7842 N Lincoln Ave
Skokie IL 60077
(800) 323-1818

International Rectifier
233 Kansas St
El Segundo CA 90245
(310) 322-3331

LSI/CSI
1235 Walt Whitman Rd
Melville NY 11747
(516) 271-0400

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

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

Porex Technologies
500 Bohannon Rd
Fairburn GA 30213
(800) 565-8777

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

Texas Instruments
PO Box 809066
Dallas TX 75380
(800) 336-5236

Times Microwave
PO Box 5039
Wallingford CT 06492
(203) 949-8400

TV Technology
5827 Columbia Pike #310
Falls Church VA 22041
(703) 998-7600

WebWeek
20 Ketchum Street
Westport CT 06880
(708) 564-1385

But the greatest new features seem to be *Acrobat Search* and *Acrobat Catalog*. I guess I have now written some 1800+ technical articles. It's getting real hard to remember when I last wrote on paleomagnetism, steam calliopes, synthetic kale, or flying car labor-of-love newsletters.

Acrobat Catalog is exceptionally good at taking a humongous pile of related documents and letting you *rapidly* and *globally* search *all* of them for keywords. Not only does Search give you keyword locations, but it even guesses how "relevant" the hits are on each found document! And there is *no limit* to how many keywords you can use! You can even search for *numbers* or text buried in your illustrations!

Full synoymns and "sounds like" features, too. With logic.

Here's how this gets done: Acrobat Catalog first "insides out" the *entire* library, alphabetizing *all* the words, along with numbers for *each* page of *each* document in which that word appears. A special .PDX file is then created holding this information.

Acrobat Search then can quickly go through this single new .PDX file looking for hits. Only the few single pages including the selected hits get exhaustively searched.

Since searching on keywords such as "an" or "the" is singularly useless, you'll usually strip down the file by removing *excluded words*. You have a choice here of a very tight and fast .PDX file with only a few keywords. Or else a larger and slower one that includes vastly more choices.

I am in the process of making all my stories globally searchable. I do hope to place the whole works on CD ROM someday soon. This should also speed up how fast I can release hard copy reprints. Search features will be added to *www.tinaja.com* shortly.

Several Acrobat hints: Always be sure to work with a Mac or Windows reader. The DOS and UNIX versions apparently are not being upgraded. At least not so far. A possible reason for this are third-party screen drivers that can't be redistributed free.

Also, the Acrobat "security" levels seem to have serious problems.

Certain uploading protocols can distort the file just enough to prevent valid users from ever gaining access. Especially XMODEM. And, if you give your user the ability to print his file, he can subvert *all* other security levels. Any time he wants to. Details in INSECURE.TXT.

I do not personally use any of the Acrobat security features.

If you want to jump into Acrobat, their *Adobe Developer's Association* has a $500 yearly developer program. This gives you the latest insider CD's with all of their latest software.

I've posted scads more info, tech docs, and great heaping bunches of Acrobat stuff at *www.tinaja.com*. I stock the Acrobat "pewter" reference manual here at my *Synergetics*.

New Tech Lit

From *Texas Instruments*, a CD on *MOS Memory Specifications*. Plus a new *Power Semiconductors Product Digest* from *International Rectifier*.

The full text of all of the FCC regs are now available on the internet as ***http://www.pls.com:8001/his/cfr.html*** Actually, the *entire* CFR or *Code of Federal Regulations* is stashed here.

With powerful search features.

Those miniature melody generators as used in greeting cards and such are distributed in quantity by *Dicker*. A melody chip source is *LSI/CSI*.

Some backgrounder info on those *VCR Plus* codes has been posted as file VCRPLUS.TXT.

Industrial Market Place is a great surplus trade journal. This is mostly mechanical stuff, but there's lots of computers and electronics as well.

WebWeek is a new net magazine by the *PCWeek* and *MACWeek* folks. Free to qualified subscribers.

Samples of porous plastics are now offered by *Porex Technologies*. The specs on low melting point alloys are in an *Indium Corporation* flyer.

A major improvement in polymer solar cells is described in *Science* for December 15, 95. Pages 1789-91.

A free coaxial cable handbook is available through *Times Microwave*. Included are detailed specs and full ap notes on most cables.

Magic sinewaves are creating a revolutionary new power electronics development. They greatly improve the efficiency and can dramatically reduce the costs of such products as induction motor speed controls, home energy efficiency improvers, electric autos, and solar inverters. This is a *billion* dollar new opportunity that's largely up for grabs.

Quite a bit has happened on the magic sinewave front in the last few weeks. I'll be happy to send a free tutorial reprint to anyone who wants one. Formal proposals and a rather detailed tutorial library are also now available to serious inquirers only. See my file MSINPROP.PDF on my web site.

Seminars, my PIC source code, and codeveloper programs are also newly available. Seminars are done in-plant or in a wilderness resort lodge.

A reminder that I do try to run a technical helpline per the *Need Help?* box. While there is no charge for this unique service, my funding and time constraints do strictly limit this to US callers only. I've also got a great new surplus catalog available. As usual, most of the mentioned sources appear in the *Names & Numbers* or *Online TV Engineering* sidebars. ✦