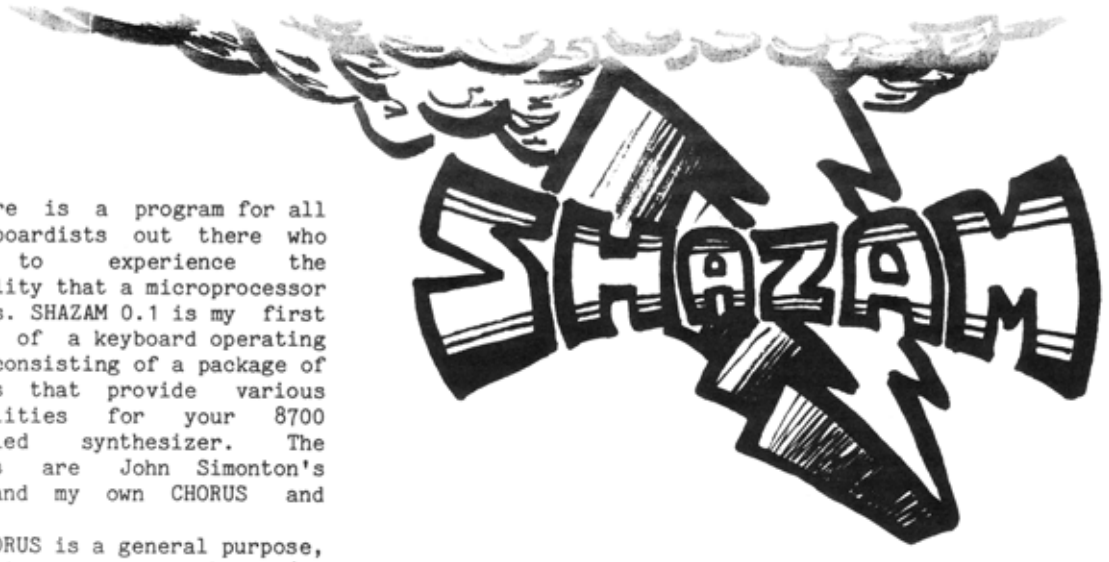

(or "Maestro, a little software, please")

By: Bob Yannes

Here is a program for all you keyboardists out there who want to experience the flexibility that a microprocessor provides. SHAZAM 0.1 is my first version of a keyboard operating system consisting of a package of programs that provide various personalities for your 8700 controlled synthesizer. The programs are John Simonton's POLY, and my own CHORUS and SPLITZ.

CHORUS is a general purpose, monophonic sequential (or circular) assignment in which successive notes are assigned to successive output channels in sequential order (sequential assignment) until all available voices are used, in which case the assignment wraps around to the beginning again (circular). The unique feature of this routine is that it allows notes to continue decaying on one channel while the new channel is playing (even if you hit the same key several times). This produces a rich pseudo-reverb or chorusing effect that greatly enhances monophonic brass and string patches. The problem with ordinary monophonic synthesizers is that if you play too fast and don't allow the envelope generators to reset, the attack portion of the envelope is bypassed which can totally ruin string sounds (ie: instead of a slow swell, your violins become percussive and go "BONG"). While the sequential assignment helps reduce this problem, the amount of reduction is dependent on how many voices you have (the more the better) and how fast you play. In general, the more voices you have, the longer you can make the final decay on each voice and still use slow attacks. CHORUS also has some other features which I'll cover in detail later.

SPLITZ is a general purpose keyboard splitting program which allows you to split the keyboard up arbitrarily into as many voices as you have (or want).While this is best utilized with larger keyboards, I still find it useful on my three octave. The important thing to note is that you are free to place the split points wherever you want and make the split sections of the keyboard as short or as long as you want (thanks to our friend, the microprocessor). A split keyboard is useful for playing polyphonically with <u>different</u> patches on each voice. I'm sure that those of you who have tried to coordinate different patches while running POLY have found it a bit difficult. The problem is that your timing must be exact when pressing the keys down, or your bass line will come out as a soprano patch and while your mind is trying to sort out what happened (and that can take some time if you're playing a complex polyphonic passage) POLY will continue to assign things wrong until you straighten it out. This isn't a fault of the algorithm, just human imprecision, but it only has to happen once to ruin a live performance. With SPLITZ, I use a brute force approach to overcome this problem. The keyboard is split up into various, non-interacting sections, with each section always being assigned to its own output channel. The program doesn't care when you play a note relative to other keys, if you play within a certain split section, that note will <u>always</u> be assigned to the same output channel. If you had a five octave keyboard, you could elect to split the keyboard up into five octaves, with each octave being assigned to its own voice. You could then patch the synthesizers up in order--channel one with a bass patch, channel two with a lead patch, channel three with a counterpoint, etc. While this is flexible and allows you to coordinate a large number of different voices, you must remember that the split up keyboard is rather small now (in the previous example, you would be limited to playing in only one octave for each voice) and even with the ability to separately transpose each voice this is a pretty big limitation, hence the program lends itself best to a few large splits (like on STRINGZ & THINGZ, you could split your keyboard to provide an octave of cellos and two octaves of violins). A popular set-up would be to give yourself an octave of synthesizer bass and two octaves of lead line patch, that way you can play your bassline without having to buy a new keyboard.

CHORUS, SPLITZ and POLY are tied together as a single package and use the 8700 keyboard to provide a tuning function, a clear function (for removing transpositions, glide or wrong channel assignments) and to select any one of the three programs (while in one program it is possible to jump to either of the other two at the touch of a button). One nice feature of this is that you can jump to any of

the other programs with no noticable glitch, hence in the middle of a chord, you can jump to sequential assignment without the synthesizer going BRAAAAK (just shift the gears, don't worry about the clutch!).

Now that you know what it can do, let's see how to use it. If you look at the program you'll find that it is written to reside in page 1 of your computer. If those of you who are familiar with the operations of the 6502 have regained consciousness, let me assure you there's no problem here. Although page 1 is dedicated to the stack, the program conveniently puts the stack out of the way and, unless you make an error in loading the program, you won't have any problems. The reason I wrote it for page 1 is to make it applicable to those who don't have the expanded memory option for the 8700 (pages 2 and 3). If you have the extra memory, feel free to load the program into page 2 or 3, the only thing you have to change is the absolute address of subroutine CLEAR in the JSR instructions at ADDR 013D and 018A and the jump at 01B9. This will simplify things somewhat. If you have to load the program into page 1, be sure to set the user stack and the monitor stack far, far away before trying to load the program (either by hand or off cassette) by changing ADDR 00FE and 00ED (the user stack pointer and monitor stack pointer, respectively) to $FF. This places the stacks at the same location starting at the bottom of page 1. Once you have the program in, the first thing you should do is save it on cassette, cause if you did make a mistake, I can almost guarantee that the stack will take a bite (byte?) out of your program.

One thing I should mention is that the MUS-1 PROM is an absolute necessity for this program. It provides all of the housekeeping as well as the entire POLY routine. Actually, MUS-1 is the most important part of the system if you intend to write any software--it's well worth the price, but remember to watch out for it's zero page variables when you're writing software (that's why SHAZAM won't fit on page zero!).

Okay, the program's in (and ostensibly correct), let's run it. First be sure to set the MUS-1 operating parameters; the Control Word at 00E8 (use $40 to test for now), the QuASH settling delay at 00E9 (I use $20) and the number of voices you have at 00EA. Incidently, I'm assuming you have an entire synthesizer for each voice, which is what true polyphony is all about. If you're using a polytonic synthesizer (ie: multioscillator monophonic) these programs won't be realized to their fullest extent, but experiment anyway. Set each of your synthesizers to the same patch for testing purposes and run the program from 0100. SHAZAM starts up running POLY. This gives you the option of tuning your system--just touch key 1 on the 8700 keyboard and while holding it, tune your oscillators. You can now play with POLY if you want--in fact it's a good idea to check out each program thoroughly before going to the next. Press key 0 on the 8700 and see that everything is cleared (the system should also clear itself when you first start running the program). Begin playing POLY and be sure that it responds to the proper number of voices and that the channels are assigned correctly (the displays should also be counting). Check the transients if you want, by changing the control word. When you are satisfied that POLY is working correctly, press the "PCH" key on the 8700. The displays should stop counting and display the number of voices. Begin playing monophonically and see that everytime you play a key, it comes out a new channel. Turn up the final decay to see what CHORUS is all about. Note that when you play the same key a number of times in a row, it is still assigned to different channels, unlike POLY which specifically avoids this. This can be very effective in creating simulated echoes by rapidly pressing and releasing the same key (especially with glide on). As I stated before, CHORUS is monophonic and provides a low-note priority. Unlike some other low-note synthesizers, pressing a lower note while holding the previous note will not result in a slur, instead the new note will be articulated and upon release of the second note, the original note will be re-articulated. Hence fully articulated, sequencer-like trills can be produced by holding down a note and rapidly pressing and releasing a lower note. A strumming effect can be achieved by pressing down a number of notes at once and rapidly releasing them in sequential order from the lowest to the highest.

When you've tired of CHORUS (hopefully it should take a while) you can return to POLY by pressing "TAPE" on the 8700 keyboard, or you can jump to SPLITZ by pressing "PCL". Upon entering SPLITZ, the program is waiting for you to enter your split points, so don't start playing yet! You have to load a number of split points that is one less than the number of voices you specified. Actually, you don't have to worry about this, the program will only let you enter that many. You enter your split points by simply playing them on the keyboard, one at a time, from lowest to highest (the displays will keep track of how many you've loaded by counting down from the number of voices). Once you've entered the last one, the next note you play will start the main part of SPLITZ running and the note will be sounded from its appropriate channel. For example, suppose you have three voices specified (at 00EA), SPLITZ is waiting for you to enter two split points. You decide you want to split the keyboard at the Cs (this will provide three independent 1-octave keyboard sections, one for each of your three voices) so first you press the second C on the keyboard, which causes the displays to show 03 for the number of voices and enters that note as the first split point (ie: all notes below that point will be assigned to channel one). Now you release that note and press the third C on the keyboard, the display decrements and that note are entered as the second split point (ie: all notes below that point, up to and including the first split point, are assigned to channel two). The program will automatically assign any remaining keys to the highest available channel (ie: any note from the second split point on up, which on a three-octave keyboard will provide an octave for channel three). After releasing your second split point, the program is ready to run, and the next note you play will automatically be sounded from it's proper channel (ie: if you now pressed low F, it will be

sounded from channel one, as it is within the channel one split region).

Verify that you can play a note within a split section and that there is no interaction or improper channel assignments, no matter when you hit the note relative to other notes you are playing in the other split regions. I should point out that while there is no priority between split sections, within a split section high-note priority prevails, so if you hit more than one note within a single split section only the highest will be sounded and, unlike CHORUS, the articulation will be slurred (ie: the note won't retrigger). To fully realize the potential of SPLITZ, change the patches (and tuning if you desire) so they are different for each voice and see how easy it is to coordinate your bass and lead, etc. Something you should bear in mind is there is no reason why the voices should be in order, pitchwise. By using the oscillator tuning knob and/or the transposition capabilities of MUS-1, you can easily make the split points which are higher on the physical keyboard, lower in frequency than those split sections below. This could come in handy if you need to play a complex bass line with a simple melody and you have more finger facility in your right hand. You could also tune each split section the same, allowing you to play the same note or close chords on different channels. Another technique you shouldn't overlook is that, with each channel patched differently, a form of limited instant patch switching can be achieved by simply changing your playing from one split section to another. The limitation is that your range within any patch is limited by the size of the split section. Perhaps some of you software wizards out there can come up with an easy way to split the keyboard into separate polyphonic sections which would overcome the monophonic (within a single split section) limitations.

If at any time you want to change the split points while running SPLITZ, simply press the "ENT" key on the 8700 and the program will jump to the beginning and await entry of the new split points. I should warn you that you must release a split point before entering a new one (the program ignores you if you hold more than one key down during the split selection) and the points must be pressed in ascending order (if you don't press them in ascending order, during play, those improper split points will cause the associated channel to be ignored). If you make a mistake, just press "ENT" when the program is running and try again.

During the development of this program, I originally had the program sound the split points as they were played, however, this feature was rejected as undesirable for live performance. If you desire to have the points sounded, you must insert a new line of code STA (NTS), Y--91 99 between 0181 and 0182--in other words, you'll have to reassemble the whole program. Good luck.

MISCELLANEOUS:

Oh yeah, while running SPLITZ, the displays will count just like POLY. Once you've tired of SPLITZ, you can return to CHORUS by pressing "PCH" or POLY by pressing "TAPE". If you want to use less than the maximum channels that you have, just enter whatever number you want to use in 00EA, the program automatically compensates. Please note that, although NTABLE can maintain 16 voices, KTABLE will only keep track of eight keys, so don't try to use more than eight channels for either CHORUS or SPLITZ (or POLY unless STGs are selected). Also, CHORUS will always play the first key pressed from channel two. If the programs don't seem to be working, look at the zero page variables and see if you can pinpoint anything that isn't what it should be, then go back to the program near the point where something went wrong and check for bad bytes. If the whole program blows up when you run it, load it back off cassette and check it closely before running it (remember, watch out for that monitor stack!). If you need to use the breakpoint routine for debugging, you can replace the first three bytes of the program with the breakpoint vector, just remember to set the stack pointer to $FF (at ADDR 00FE) before running the program. If at any time you want to select glide or transpose a channel, stop the program (RESET) and load the desired transposition factor in the appropriate location in TTBL (00C0-00CF). For example, if you wanted to turn the glide on for the first two channels, load 00CF and 00CE with $80. Before running the program, set the status register at 00FF to 00 to assure that HEX addition will take place, then run the program from 0111 (this assumes that the program has been run at least once from the beginning and NTS is loaded with the proper value). If you forget and run the program from the start, INIT will clear TTBL and you'll have to start over. Likewise, if you press key 0 on the 8700 during POLY, TTBL will also be erased, which is an easy way to turn glide off.

For you programmers out there, I'd like to point out a little trick which simplified these programs immensely. It involves the use of indirect, indexed addressing and redefining the start of NTABLE with respect to the number of voices used. Normally a program would keep track of how many voices were filled by counting down from the number of voices specified. Upon reaching zero, we know that they are all done so we branch or something. Unfortunately, NTABLE (and therefore the output channels), are in descending order starting from 00DF to 00D0, so the voice counter cannot be used as a storage pointer to NTABLE. This means we need two loops, one counting down from the number of voices to zero, and one counting down from the top of NTABLE to be used as a pointer. This really gets messy when we have to keep track of those two numbers while jumping to subroutines that blow up the X, Y and A registers. Instead of using two loops, I simply redefined the start of NTABLE based on the number of voices by subtracting the number of voices from $DF (the highest position in NTABLE) and storing it in NTS (NTABLE Start). The rest of the program uses this reference for NTABLE via indirect addressing (which is why location NTS+1 must be set to 00).

NOTE: For those of you who want to retain split points while jumping back and forth between programs, simply change ADDR 0136 to $53 and ADDR 015B to $2E. This causes these instructions to become "BEQ PERF" instead of "BEQ SPLITZ". Now any time you jump to

the split program, it will immediately start running using the previously loaded split points. If you haven't loaded any yet, or if you wish to change the previous split points, just press "ENT".

CONCLUSION:

If you're wondering why I call this program version 0.1, it's simply because I have much bigger plans for the future. A fully polyphonic version of CHORUS is under development (well, it almost works) which will certainly become my favorite assignment algorithm as it combines all the best stuff of POLY with all the best stuff of CHORUS--and none of the limitations! If you think SHAZAM 0.1 is enough to satisfy you, just wait! Remember, the advantage of the 8700 is that you are free to produce any kind of bizarre algorithm you want, tailored to your specifications. I can think of some neat algorithms which SHAZAM doesn't even touch on. How about a channel assignment algorithm in which whatever channel the full keyboard is assigned to is determined by the 8700 keyboard. You could attach a number of differently patched synthesizers and switch patches instantly by selecting another channel, and this time you have a full keyboard (monophonic) to play with. Granted this is hardly cost effective compared to a true programmable synthesizer (like Oberheim's OB-1) but if you have the hardware already, it could be a nice change of pace from polyphonic playing! In the end, we intend to have all of these routines linkable so that, for example, a sequencer can be fed to the sequential assignment to give CHORUSing to a digital sequence. This could form a complex operating system similar in many respects to operating systems on big computer systems. What hath PAIA wrought?

SHAZAM 0.1

A Keyboard Operating System For The PAIA 8700 Computer

By: Bob Yannes

October 1978

```
ADDR   CODE         LABEL    INSTRUCTION     COMMENTS

0100   A2 FF                 LDX #$FF        GET THE STACK
0102   9A                    TXS             OUT OF THE WAY.
0103   A9 DF                 LDA #$DF        GET NTABLEMAX POSITION...
0105   38                    SEC             PREPARE TO SUBTRACT...
0106   E5 EA                 SBC OUTS        WELL, SUBTRACT THE NUMBER OF VOICES
0108   85 99                 STA NTS         THIS IS THE START OF NTABLE NOW.
010A   A9 00                 LDA #0          ASSURE ADDR HIGH IS ZERO (PAGE)
010C   85 9A                 STA NTS+1       FOR USE AS INDIRECT ADDR.
010E   20 21 0D     OPTN     JSR INIT        WIPE'EM OUT!
0111   20 71 0D     PLOOP    JSR POLY        ASSIGN NOTES ACCORDING TO POLY.
0114   20 C3 0D              JSR TRNGN       DO TRANSIENTS IF SELECTED.
0117   20 2B 0D              JSR NOTE        PLAY THE NOTES.
011A   A5 BF                 LDA CLK         BELLS AND
011C   8D 20 08              STA DISPLAY     WHISTLES.
011F   20 00 0F              JSR DECODE      SCAN 8700 KYBD.
0122   C9 01                 CMP #1          IS IT TUNE?
0124   90 E8                 BCC OPTN        NO, IT'S LESS--MUST BE CLEAR.
0126   D0 07                 BNE MORE        NO, GO SEE WHAT IT WAS.
0128   A0 5C                 LDY #2nd C      YES, GET NOTE TO TUNE WITH...
012A   20 52 0D              JSR FILL        PUT IN ALL CHANNELS...
012D   F0 E2                 BEQ PLOOP       AND PLAY IT.
012F   C9 14        MORE     CMP #"PCH"      IS IT PCH?
0131   F0 06                 BEQ CHORUS      YES, GO TO CHORUS,
0133   C9 15                 CMP #"PCL"      OKAY, THEN IS IT PCL?
0135*  F0 32                 BEQ SPLITZ      YES, GO TO SPLITZ.
0137   D0 D8                 BNE PLOOP       NO, TWAS NONE OF THEM, KEEP ON POLY.

0139   A5 EA        CHORUS   LDA OUTS        GET THE NUMBER OF AVAILABLE VOICES.
013B   85 98                 STA COUNT       USE AS COUNTER/POINTER.
013D   20 BC 01     IN       JSR CLEAR       TURN OFF ALL GATES.
0140   A5 E7                 LDA KTBLMAX     GET LOWEST NOTE.
0142   F0 0C                 BEQ OUT         IF ZERO, SKIP ASSIGNMENT.
0144   C5 97                 CMP OLDKEY      IS IT THE SAME KEY YOU JUST HAD?
0146   F0 04                 BEQ SAME        YES, KEEP OLD CHANNEL ASSIGNMENT.
0148   C6 98                 DEC COUNT       NO, GET NEW CHANNEL ASSIGNMENT.
014A   F0 04                 BEQ OUT         IF ZERO, SKIP ASSIGNMENT.
014C   A4 98        SAME     LDY COUNT       GET POINTER.
```

```
014E    91 99              STA (NTS), Y    ASSIGN VOICE TO CHANNEL Y.
0150    85 97      OUT     STA OLDKEY      ALSO, STORE IT FOR LATER.
0152    20 2B 0D           JSR NOTE        PLAY THE NOTE.
0155    20 00 0F           JSR DECODE      SCAN 8700 KYBD.
0158    C9 15              CMP #"PCL"      IS IT PCL?
015A*   F0 0D              BEQ SPLITZ      YES, GO TO SPLITZ.
015C    C9 16              CMP #"TAPE"     IS IT TAPE?
015E    F0 B1              BEQ PLOOP       YES, GO TO POLY.
0160    A5 98              LDA COUNT       GET COUNT.
0162    F0 D5              BEQ CHORUS      IF ZERO, START OVER.
0164    8D 20 08           STA DISPLAY     IF NOT, GIVE US A LOOK...
0167    D0 D4              BNE IN          AND KEEP ON CHORUS.
0169    A5 EA      SPLITZ  LDA OUTS        GET THE NUMBER OF AVAILABLE VOICES.
016B    85 98              STA COUNT       USE AS COUNTER/POINTER.
016D    85 97      SELECT  STA OLDKEY      STORE THE PRESENT NOTE FOR LATER.
016F    20 2B 0D           JSR NOTE        REFRESH QUASH AND LOAD KTABLE.
0172    A5 E7              LDA KTBLMAX     GET LOWEST NOTE.
0174    F0 F7              BEQ SELECT      IF ZERO, TRY AGAIN.
0176    A4 97              LDY OLDKEY      GET LAST NOTE.
0178    D0 F3              BNE SELECT      IF NOT ZERO, YOU'RE STILL HOLDING IT.
017A    A4 98              LDY COUNT       IT'S A NEW NOTE--GET COUNTER/POINTER.
017C    8C 20 08           STY DISPLAY     SHOW THE COUNT.
017F    99 9A 00           STA SPLIT, Y    STORE THE NOTE AS SPLIT POINT Y.
0182    C6 98              DEC COUNT       NEXT SPLIT POINT.
0184    D0 E7              BNE SELECT      GO BACK UNTIL ALL SPLIT POINTS ARE IN.
0186    A9 FF              LDA #$FF        DUMMY SPLIT POINT...
0188    85 9B              STA SPLITMIN    STORED AT END OF TABLE.
018A    20 BC 01   PERF    JSR CLEAR       ALL SPLITS ARE IN, TURN OFF ALL GATES.
018D    A2 08              LX #8           PREPARE TO TEST KTABLE.
018F    B5 DF      NXTNOT  LDA KTBL, X     GET NOTE X FROM IT.
0191    F0 0C              BEQ NXT2        IF ZERO, DO NEXT.
0193    A4 EA              LDY OUTS        GET COUNTER/POINTER.
0195    D9 9A 00   NXTO    CMP SPLIT, Y    COMPARE NOTE TO SPLIT POINT Y.
0198    90 03              BCC NXT1        IF LESS THAN, GO ASSIGN IT.
019A    88                 DEY             NEXT SPLIT POINT.
019B    D0 F8              BNE NXTO        KEEP ON UNTIL YOU'VE CHECKED ALL.
019D    91 99      NXT1    STA (NTS), Y    ASSIGN NOTE TO CHANNEL Y.
019F    CA         NXT2    DEX             NEXT KEYBOARD ENTRY.
01A0    D0 ED              BNE NXTNOT      KEEP ON UNTIL ALL KEYBOARD DONE.
01A2    20 2B 0D           JSR NOTE        PLAY THE NOTES.
01A5    A5 BF              LDA CLK         MORE BELLS AND
01A7    8D 20 08           STA DISPLAY     WHISTLES.
01AA    20 00 0F           JSR DECODE      SCAN 8700 KYBD.
01AD    C9 14              CMP #"PCH"      IS IT PCH?
01AF    F0 88              BEQ CHORUS      YES, GO TO CHORUS.
01B1    C9 13              CMP #"ENT"      IS IT ENT?
01B3    F0 B4              BEQ SPLITZ      YES, GO GET NEW SPLIT POINTS.
01B5    C9 16              CMP #"TAPE"     IS IT TAPE?
01B7    D0 D1              BNE PERF        NO, KEEP ON PLAYING SPLITZ.
01B9    4C 11 01           JMP PLOOP       YES, GO TO POLY.

01BC    A4 EA      CLEAR   LDY OUTS        GET POINTER/COUNTER.
01BE    B1 99      NANO    LDA (NTS), Y    GET NOTE BEING PLAYED ON CHANNEL Y.
01C0    29 3F              AND #$3F        TURN OFF ITS GATE...
01C2    91 99              STA (NTS), Y    AND PUT IT BACK.
01C4    88                 DEY             NEXT CHANNEL.
01C5    D0 F7              BNE NANO        GO BACK UNTIL DONE.
01C7    60                 RTS             RETURN.
```

LOCATION	NAME	PURPOSE
0097	OLDKEY	CONTAINS PREVIOUS NOTE FOR COMPARISON W/NEW NOTE.
0098	COUNT	VOICE COUNTER/POINTER.
0099	NTS	START OF NTABLE FOR GIVEN NUMBER OF VOICES.
009A	NTS+1	(=0) SPECIFIES ZERO PAGE FOR INDIRECT ADDRESS.
009B	SPLITMIN	DUMMY SPLIT POINT, ASSURES HIGHEST NOTE ASSIGNED TO LAST CHANNEL.
009C-00A2	SPLIT	TABLE OF SPLIT POINTS.

*See text.