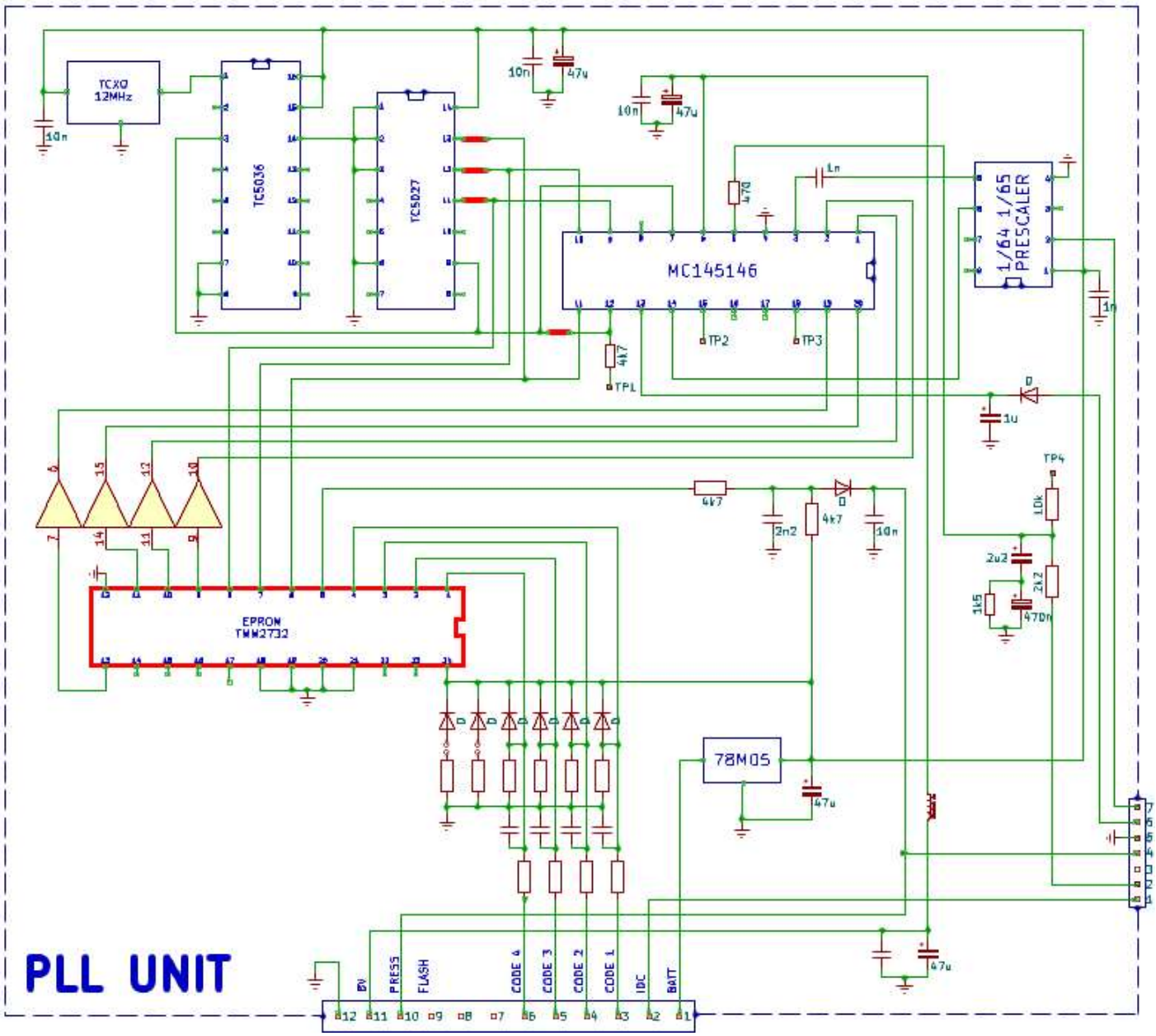
Changes to the PLL unit

The goal of the PLL controller design is to retain the Kyodo design as much as possible. Minimal intervention in the existing design was sought. The EPROM is completely replaced by a piggyback module whose pinning is the same size as the EPROM and therefore fits directly onto the EPROM's IC socket.



To eliminate the existing EPROM readout, only four changes are needed to the PCB layout. Three traces are cut around the TC5027. The traces leading to the EPROM remain intact. Unfortunately, the trace on the MC145146 runs on the component side. Therefore, cutting the trace is not possible. To free pin 12 of the trace, pin 12 of the MC145146 is cut just above the PCB and raised. A wire from the piggyback is connected to the loose pin to restore its function.

The

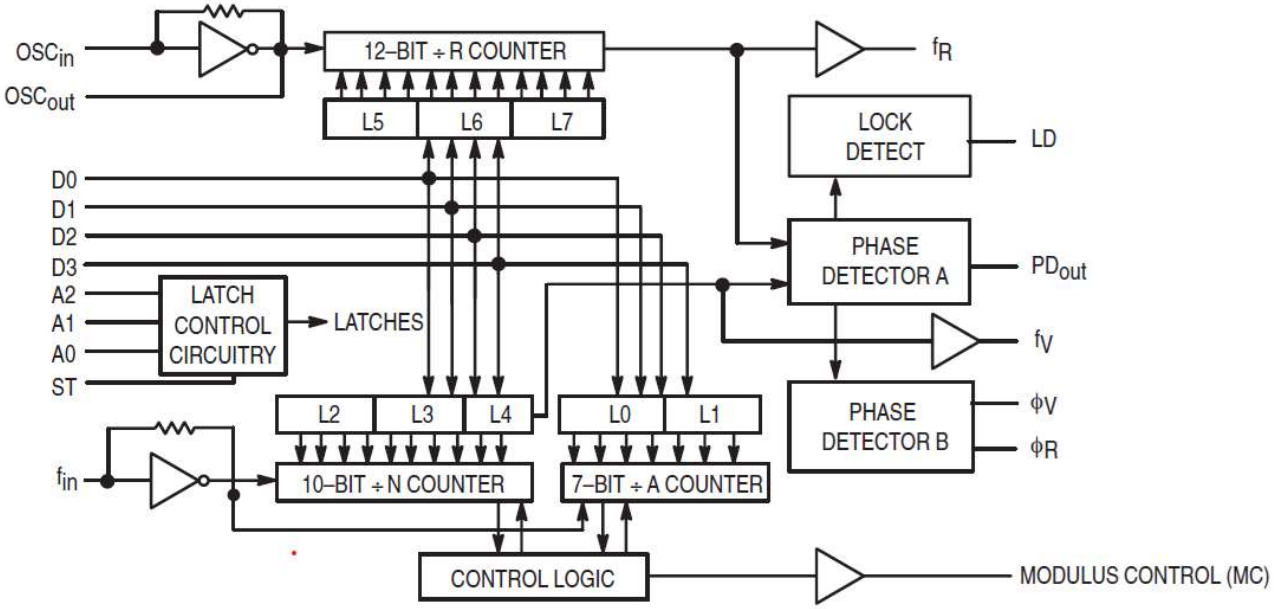


figure above is taken from the datasheet for the MC145146 "frequency synthesizer." The synthesizer is constructed with three counters. These are:

- R-counter: Referense counter. This 12-bit divider is used to divide the crystal-controlled reference frequency by the minimum grid size.
- N-counter: Normal counter. This 10-bit divider is used to divide the VFO frequency to be controlled by the reference frequency from the R-counter.
- A-counter: Alternate counter. This 7-bit divider controls the switching of the pre-scaler between 64 and 65.

For a 6250 Hz grid and a 12 MHz reference frequency, the R-counter should be divided by:

```
12.000.000 / 6250 = 1920
```

This is the dividend for the R counter. The diagram shows a 64/65 prescaler. The prescaler outputs:

```
145.000.000 / 64 = 2.265.625
```

This is divided by the grid frequency:

```
2.265.625 / 6250 = 362,5
```

The first value of the dividend has been found, namely 362. This is the value for the N-counter. The division has created a value after the decimal point. To correct this, the A-counter is used. First, the result of the division is converted back to whole numbers:

```
(362,5 – 362) x 64 = 0,5 x 64 = 32
```

The result is placed in the A-counter. The A-counter controls the pre-scaler. This means that for 32 counting cycles of the N-counter, the pre-scaler divides by 65. Then, the pre-scaler divides by 64. So:

```
        32 x 65 x 6250        =  13.000.000
(362 – 32) x 64 x 6250        = 132.000.000
                                ------------ +
13.000.000 + 132.000.000      = 145.000.000 = 145 MHz
```

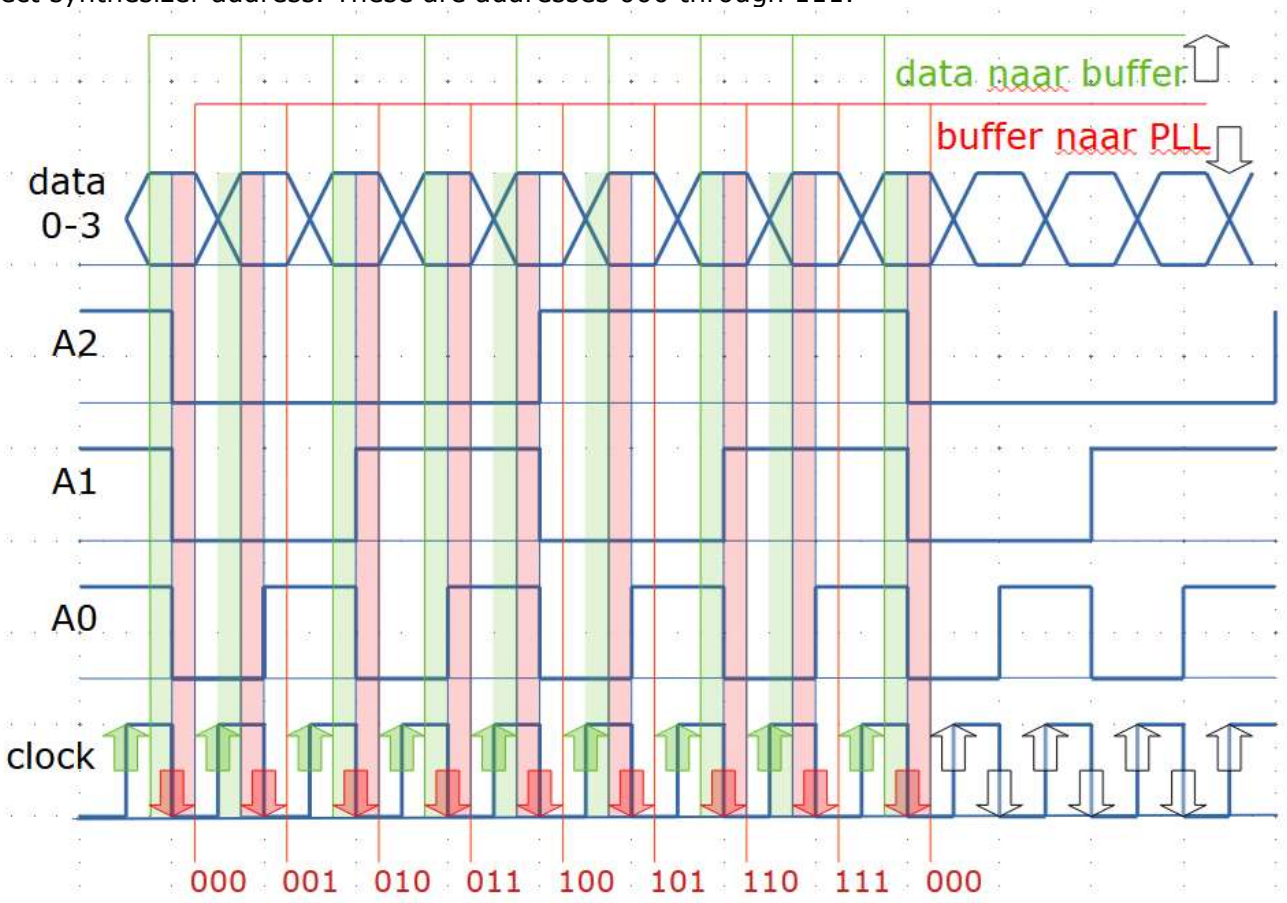
The divisors for the synthesizer are converted to four bits and hexadecimal.

R-counter 1920	0x780	L5 = 0x00	L6 = 0x08	L7 = 0x07
N-counter 362	0x16A	L2 = 0x0A	L3 = 0x06	L4 = 0x01
A-counter 32	0x20	L0 = 0x00	L1 = 0x02	

In the original design, the EPROM is programmed by frequency. The correct addresses are read by setting the channel selector. The counters in the synthesizer are 4 bits, namely bit 0 through bit 3.

	L0	L1	L2	L3	L4	L5	L6	L7
145,00000 MHz	00	02	0A	06	01	00	08	07
145,00625 MHz	01	02	0A`	06	01	00	08	07
145,01250 MHz	02	02	0A	06	01	00	08	07
145,01875 MHz	03	02	0A	06	01	00	08	07

The PLL unit's schematic uses a divide-by-two mechanism to convert the values of L0 through L7 to the correct synthesizer address. These are addresses 000 through 111.



Address lines A0 through A2 are formed by dividing the clock (ST) frequency by two. The count of the two dividers is used to select both the EPROM address and the synthesizer counters. A3 of the EPROM is used to switch between Tx and Rx (PTT). In the MC145146 datasheet, the clock is designated as ST (strobe). A buffer is used to read the address and data from L0 through L7. This prevents glitching of address and data to counters. The buffer opens on the rising edge of the clock. Green arrow up. The address and data for the synthesizer are now in the buffer. During the falling edge of the clock, the buffer closes and the contents are stored. Red arrow down. The buffer contents are also placed in the addressed counter.

Based on the above information, it is possible to compile the data for the big bag by combining the counter data and the counter addresses. The counters are addressed via A0 through A2 on the synthesizer and A0 through A2 on the EPROM. From the big bag, it is compiled as follows. The byte per counter consists of 4 bits (D0 through D3) of counter data and three bits (D4 through D6) of counter addresses.

	Adresslines EPROM			places in de biggybag							
	A2	A1	A0	adreslijnen				counter data			
				D7	D6	D5	D4	D3	D2	D1	D0
Counter 0	0	0	0	x	0	0	0	y	y	y	y
Counter 1	0	0	1	x	0	0	1	y	y	y	y
Counter 2	0	1	0	x	0	1	0	y	y	y	y
Counter 3	0	1	1	x	0	1	1	y	y	y	y
Counter 4	1	0	0	x	1	0	0	y	y	y	y
Counter 5	1	0	1	x	1	0	1	y	y	y	y
Counter 6	1	1	0	x	1	1	0	y	y	y	y
Counter 7	1	1	1	x	1	1	1	y	y	y	y

Combining the previous calculations with the synthesizer's composite addresses completes the code for the piggy bag and its control. The "0x" indicates that it is a hexadecimal number.

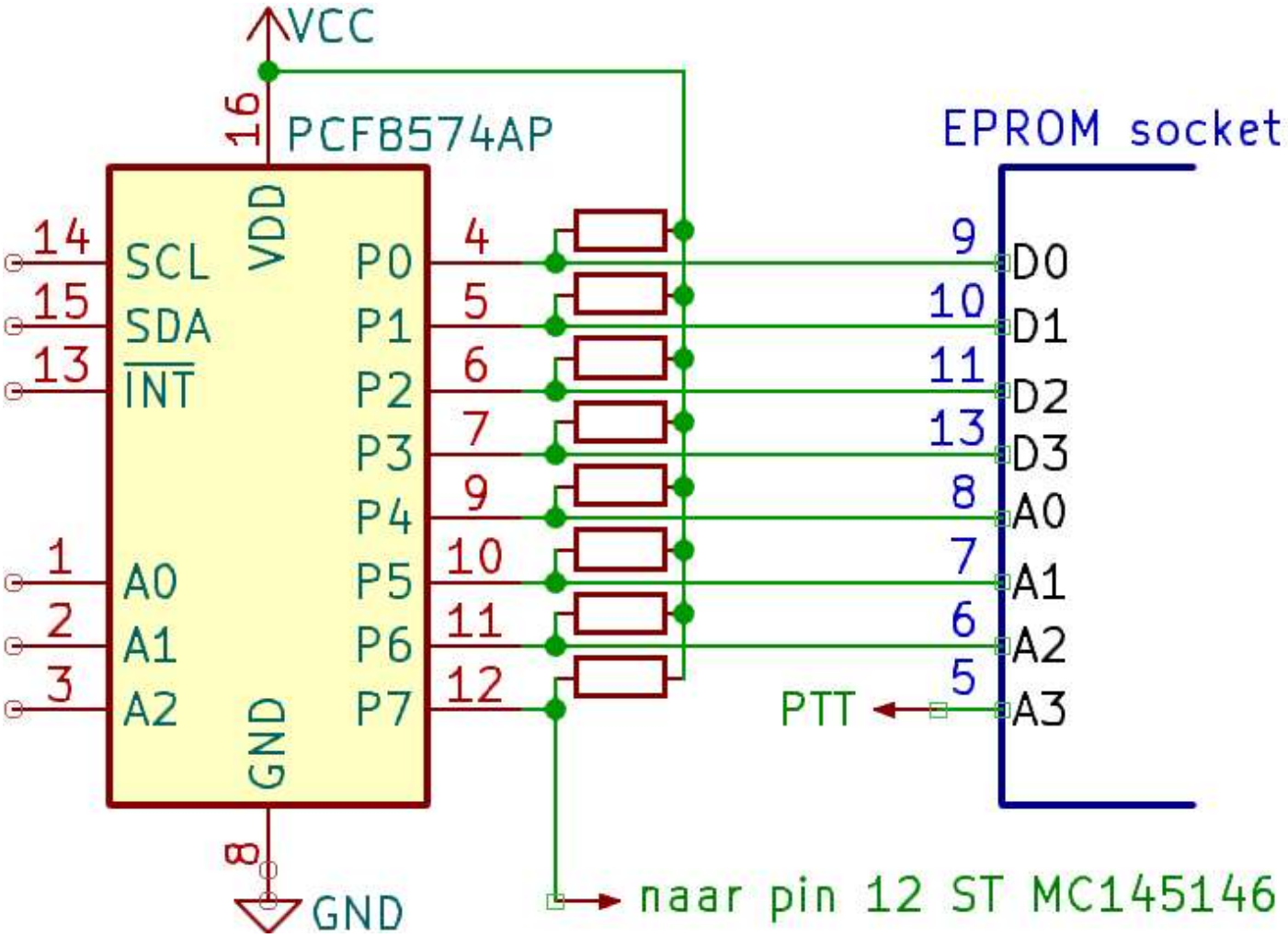
Adressing	L0	L1	L2	L3	L4	L5	L6	L7
	0x0y	0x1y	0x2y	0x3y	0x4y	0x5y	0x6y	0x7y

frequentie	L0	L1	L2	L3	L4	L5	L6	L7
145,00000 MHz	0x00	0x02	0x0A	0x06	0x01	0x00	0x08	0x07
145,00625 MHz	0x01	0x02	0x0A	0x06	0x01	0x00	0x08	0x07
145,01250 MHz	0x02	0x02	0x0A	0x06	0x01	0x00	0x08	0x07
145,01875 MHz	0x03	0x02	0x0A	0x06	0x01	0x00	0x08	0x07

Both data combined makes:

145,00000 MHz	0x00	0x12	0x2A	0x36	0x41	0x50	0x68	0x77
145,00625 MHz	0x01	0x12	0x2A`	0x36	0x41	0x50	0x68	0x77
145,01250 MHz	0x02	0x12	0x2A	0x36	0x41	0x50	0x68	0x77
145,01875 MHz	0x03	0x12	0x2A	0x36	0x41	0x50	0x68	0x77

Bit 7 (x D7) is used in the program to clock the data into the synthesizer.



To replace the EPROM, a so-called "piggy bag" is installed with an I2C 8-bit IO expander, the PCF8574. The IO expander connects to the microprocessor via SCL/SDA/INT. Because the IO expander has open-drain outputs, these are brought to Vcc via pull-up resistors. The software places the correct address and data information on the IO expander. These are 4 data bits and 3 address bits. This data is clocked to the synthesizer via software via data bit P7 of the IO expander by briefly pushing it high and then low again.

The assembly-based software uses labels. Because the PLL data is dynamic, it is placed in the microprocessor's RAM. These labels are:

```
.equ PLL_TxRAM0         = 0x0100
.equ PLL_TxRAM1         = 0x0101
.equ PLL_TxRAM2         = 0x0102
.equ PLL_TxRAM3         = 0x0103
.equ PLL_TxRAM4         = 0x0104
.equ PLL_TxRAM5         = 0x0105
.equ PLL_TxRAM6         = 0x0106
.equ PLL_TxRAM7         = 0x0107
;
.equ I2C_PLL_addr       = 0x4E
.equ I2C_segment        = 0x70
.equ I2C_display        = 0x72
```

Communication to the IO expander on the piggy bag is via the I2C bus. We've chosen to use the full start/address/data/stop cycles for each transmission via the I2C bus.

```
Tx_2_PLL:                       ; Tx_2_PLL verzorgt het overschakelen van Rx naar Tx door het
                                ; programmeren van synthesizer.
call chk_4_write_EEPROM         ; Bij het indrukken van de PTT wordt deze subroutine gebruikt om te
                                ; checken of er verschillen zijn tussen de gegevens in het werkgeheugen
                                ; en de inhoud van de EEPROM. Bij een eventueel verschil wordt de inhoud
                                ; van de EEPROM bijgewerkt. Hierdoor wordt de laatst gebruikte
```

```
                                ; frequentie en instelling ook na het uitschakelen van de transceiver
                                ; bewaard.
call Upper_4_PLL                ; Om programmageheugen te sparen en om fouten in de programmatuur te
                                ; mitigeren wordt de inhoud van L4 t/m L7 met een aparte subroutine in
                                ; de synthesizer geladen. Deze subroutine wordt ook gebruikt voor
                                ; Txrep_2_PLL, Rx_2_PLL en Rxrep_2_PLL. Zie hiervoor verder in dit
                                ; document.
                                ;
call    I2C_start               ; Start de I2C controller
ldi     r16,    I2C_PLL_addr    ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address             ; Voer met adres in in I2C controller
lds     r16,    PLL_TxRAM3      ; Laad de data en adres in voor L3 van de synthesizer
mov     r17,    r16             ; Stel de inhoud van register 16 veilig in register 17
call    I2C_data                ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                ; Stop de I2C controller
call    Prg_PLL2                ; Deze subroutine wordt gebruikt om de data in de pcf8574 over te
                                ; brengen naar de synthesizer.  Zie verderop in dit document.
                                ;
call    I2C_start               ; Start de I2C controller
ldi     r16,    I2C_PLL_addr    ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address             ; Voer met adres in in I2C controller
lds     r16,    PLL_TxRAM2      ; Laad de data en adres in voor L2 van de synthesizer
mov     r17,    r16             ; Stel de inhoud van register 16 veilig in register 17
call    I2C_data                ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                ; Stop de I2C controller
call    Prg_PLL2                ; Deze subroutine wordt gebruikt om de data in de pcf8574 over te
                                ; brengen naar de synthesizer.  Zie verderop in dit document.
                                ;
call    I2C_start               ; Start de I2C controller
ldi     r16,    I2C_PLL_addr    ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address             ; Voer met adres in in I2C controller
lds     r16,    PLL_TxRAM1      ; Laad de data en adres in voor L1 van de synthesizer
mov     r17,    r16             ; Stel de inhoud van register 16 veilig in register 17
call    I2C_data                ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                ; Stop de I2C controller
call    Prg_PLL2                ; Deze subroutine wordt gebruikt om de data in de pcf8574 over te
                                ; brengen naar de synthesizer.  Zie verderop in dit document.
                                ;
call    I2C_start               ; Start de I2C controller
ldi     r16,    I2C_PLL_addr    ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address             ; Voer met adres in in I2C controller
lds     r16,    PLL_TxRAM0      ; Laad de data en adres in voor L0 van de synthesizer
mov     r17,    r16             ; Stel de inhoud van register 16 veilig in register 17
call    I2C_data                ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                ; Stop de I2C controller
call    Prg_PLL2                ; Deze subroutine wordt gebruikt om de data in de pcf8574 over te
                                ; brengen naar de synthesizer.  Zie verderop in dit document.
                                ;
ret                             ; Keer terug naar het hoofdprogramma.
```

Prg_PLL2 handles the transfer of the PLL data via the buffer to the synthesizer's counters. Setting bit 7 makes the buffer transparent on the rising edge of the bit. Transfer to the counters is accomplished by resetting bit 7 on the falling edge of the bit. This program is used in several places in the script.

```
Prg_PLL2:                       ; Prg_PLL2 wordt door meerdere subroutines gebruikt. Dit zijn Tx_2_PLL,
                                ; Txrep_2_PLL, Rx_2_PLL en Rxrep_2_PLL. Het gebruik van deze toepassing
                                ; bespaart programmageheugen en voorkomt programmeerfouten omdat het
                                ; maar op één plaats wordt gebruikt.
                                ;
call    I2C_start               ; Start de I2C controller
ldi     r16,    I2C_PLL_addr    ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address             ; Voer met adres in in I2C controller
mov     r16,    r17             ; Lees het veilig gestelde register 17, in register 16
ori     r16,    0x80            ; Set bit 7 om de adres- en databuffer in de synthesizer te openen
                                ; De "rising edge" opent de buffer.
call    I2C_data                ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                ; Stop de I2C controller
                                ;
call    I2C_start               ; Start de I2C controller
ldi     r16,    I2C_PLL_addr    ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address             ; Voer met adres in in I2C controller
mov     r16,    r17             ; Lees het veilig gestelde register 17, in register 16, bit 7 wordt
                                ; automatisch gereset. De 'falling edge" verplaatst de inhoud van de
                                ; databuffer naar de geadresseerde counters.
call    I2C_data                ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                ; Stop de I2C controller
                                ;
ret                             ; Keer terug naar het aanroepprogramma.
```

```
Upper_4_PLL:
call    I2C_start                   ; Start de I2C controller
ldi     r16,    I2C_PLL_addr        ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address                 ; Voer met adres in in I2C controller
lds     r16,    PLL_TxRAM7          ; Laad de data en adres in voor L7 van de synthesizer
mov     r17,    r16                 ; Stel de inhoud van register 16 veilig in register 17
call    I2C_data                    ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                    ; Stop de I2C controller
call    Prg_PLL2                    ; Deze subroutine wordt gebruikt om de data in de pcf8574 over te
                                    ; brengen naar de synthesizer.  Zie verderop in dit document.
                                    ;
call    I2C_start                   ; Start de I2C controller
ldi     r16,    I2C_PLL_addr        ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address                 ; Voer met adres in in I2C controller
lds     r16,    PLL_TxRAM6          ; Laad de data en adres in voor L6 van de synthesizer
mov     r17,    r16                 ; Stel de inhoud van register 16 veilig in register 17
call    I2C_data                    ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                    ; Stop de I2C controller
call    Prg_PLL2                    ; Deze subroutine wordt gebruikt om de data in de pcf8574 over te
                                    ; brengen naar de synthesizer.  Zie verderop in dit document.
                                    ;
call    I2C_start                   ; Start de I2C controller
ldi     r16,    I2C_PLL_addr        ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address                 ; Voer met adres in in I2C controller
lds     r16,    PLL_TxRAM5          ; Laad de data en adres in voor L5 van de synthesizer
mov     r17,    r16                 ; Stel de inhoud van register 16 veilig in register 17
call    I2C_data                    ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                    ; Stop de I2C controller
call    Prg_PLL2                    ; Deze subroutine wordt gebruikt om de data in de pcf8574 over te
                                    ; brengen naar de synthesizer.  Zie verderop in dit document.
                                    ;
call    I2C_start                   ; Start de I2C controller
ldi     r16,    I2C_PLL_addr        ; Lees het adres in van de pcf8574 op de piggybag
call    I2C_address                 ; Voer met adres in in I2C controller
lds     r16,    PLL_TxRAM4          ; Laad de data en adres in voor L4 van de synthesizer
mov     r17,    r16                 ; Stel de inhoud van register 16 veilig in register 17
call    I2C_data                    ; Start de dataoverdracht naar de pcf8574 op piggybag
call    I2C_stop                    ; Stop de I2C controller
call    Prg_PLL2                    ; Deze subroutine wordt gebruikt om de data in de pcf8574 over te
                                    ; brengen naar de synthesizer.  Zie verderop in dit document.
                                    ;
ret                                 ; Keer terug naar het aanroepprogramma.
```