

RD 2000 Manager 2.30 StreamByter Inbound

Red Heron Music

March 2025

Internal Messages

Mo M1 M2 M3 M4 M5 M6

F0 58 58 00 xx yy Outbound program number 1-300, 0 = Name
load inactive, layout → output

F0 58 58 01 xx Inbound increment trigger # abandoned?

F0 58 58 02 .. Inbound Load name format – details in module,
input → layout

F0 58 58 03 skipped?

F0 58 58 04 xx Outbound – store current PC channel, layout to
output

F0 58 58 05 xx Inbound – on load, trigger current PC MIDI
channel, input → layout

F0 00 01 7E 77 00 xx – Wrap around trigger

IF LOAD

Alias G0 CurPgm # 1-\$300, 0 = inactive

Alias P0 PTemp1 # Precision for more accuracy on multiply

i00 - i0F Store Zone status bits for mixer buttons

Alias i12 Temp2

Alias i13 Temp3

Alias i14 Temp4

Alias i15 Temp5

Alias i16 Temp6

J00 - J02 - mixer evaluation section

Alias J03 TestFlag # Mfx / Trem load module

Define True 1

Define False 0

Ass CurPgm = 0 # inactive

```
Snd F0 58 58 05 01 F7 # Retrieve Program Change Channel value
Snd F0 58 58 05 00 F7 +D200
```

END

```
IF M0 == F0 41 10 00
    IF M04 == 00 75 12
```

Name Load Module

Variables

PTemp1, Temp2, Temp3, Temp4, Temp5, Temp6

CurPgm

#

Insert a value to store in correct picker location - based on CurPgm

Insert the Program Bank and Program numbers for display

RQ - Name

```
#F0 41 10 00 00 75 11 10 00 00 00 00 00 00 10 Cs F7
```

DT - Name

```
# F0 41 10 00 00 75 12 10 00 00 00 N1 N2 N3 N4 N5 N6 N7 N8 N9
NA NB NC ND NE NF N10 Cs F7
```

M0 M1 M2 M3 M4 M5 M6 M7 M8 M9 MA MB

Reformat

```
# F0 58 58 02 V1 V2 A - nn nn sp N1 N2 N3 N4 N5 N6 N7 N8 N9
NA NB NC ND NE NF N10 Cs F7
```

V = 2b, S = 21b

```
IF M07 == 10 00 00 00
```

```
IF G0 != 0 # Only execute when name load in progress
```

```
IF ML == $29 # 13 bytes message structure + 16 alfa bytes
```

Calculate MD control MIDI value 0 - 16383 in 300 ticks

Mat PTemp1 = CurPgm - 1

Mat PTemp1 = \$16383 * PTemp1

Mat PTemp1 = PTemp1 / \$299 # Nr of ticks (programs) -1

Split to two bytes

Mat Temp2 = PTemp1 / 80

Mat Temp3 = PTemp1 % 80

G01 = A-01

Calculate A-## for display

Letter (CurPgm - 1) / 20

Ass Temp4 = CurPgm

Mat Temp4 = Temp4 - 1

Mat Temp4 = Temp4 / \$20

Digit 1 = (((CurPgm - 1) Mod 20) + 1) / 10

Mat Temp5 = CurPgm - 1

Mat Temp5 = Temp5 % \$20

Mat Temp5 = Temp5 + 1

Mat Temp5 = Temp5 / \$10

Digit 0 = CurPgm Mod 10

Mat Temp6 = CurPgm % \$10

Convert to alfa

Mat Temp4 = Temp4 + \$65

Mat Temp5 = Temp5 + \$48

Mat Temp6 = Temp6 + \$48

Ass M01 = 58 58 02 Temp2

Ass M05 = Temp3 Temp4 \$45 Temp5

Ass M09 = Temp6 \$32

END

END

END

END Name Load

Mixer Read Logic - Rev 2

Build this array, then evaluate

I00 I01 I02 I03 ...

Z1I Z1E Z2I Z2E ...

Variables I00 - I0F - **Warning Addressed Indirectly**

J00, J01, J02

some of the first SB code I ever wrote, years ago, prolly could be improved, but it works.

IF M07 == 10 00 20 0F #Z1

ASS I00 = M0B

END

IF M07 == 10 00 28 0F #Z2

ASS I02 = M0B
END

IF M07 == 10 00 30 0F #Z3
ASS I04 = M0B
END

IF M07 == 10 00 38 0F #Z4
ASS I06 = M0B
END

IF M07 == 10 00 50 0F #Z5
ASS I08 = M0B
END

IF M07 == 10 00 58 0F #Z6
ASS I0A = M0B
END

IF M07 == 10 00 60 0F #Z7
ASS I0C = M0B
END

IF M07 == 10 00 68 0F #Z8
ASS I0E = M0B
END

External

IF M07 == 10 00 40 05 #Z1
ASS I01 = M0B
END

IF M07 == 10 00 42 05 #Z2
ASS I03 = M0B
END

IF M07 == 10 00 44 05 #Z3
ASS I05 = M0B
END

IF M07 == 10 00 46 05 #Z4
ASS I07 = M0B
END

```
IF M07 == 10 00 70 05 #Z5
    ASS I09 = M0B
END
```

```
IF M07 == 10 00 72 05 #Z6
    ASS I0B = M0B
END
```

```
IF M07 == 10 00 74 05 #Z7
    ASS I0D = M0B
END
```

```
# Z8 skipped here, used as trigger for evaluation below
# Z8 external needs to be first / last in scan
```

```
# Final external status triggers evaluation
```

```
IF M07 == 10 00 76 05
    ASS I0F = M0B
```

```
ASS J00 = 0 # Pointer to zone data
ASS J01 = 1 # Pointer to second byte
ASS J02 = 1 # Zone evaluated
```

```
IF J00 < 18 +Loop
```

```
    # IJ00, IJ01
    # 00 00 = Off
    # 00 01 = External
    # 01 00 = Internal
    # 01 01 = Both
```

```
    IF IJ00 == 00
        IF IJ01 == 00
            SND F0 08 08 J02 00 F7 # Off
        END
        IF IJ01 == 01
            SND F0 08 08 J02 55 F7 # Ext
        END
    END
```

```
    IF IJ00 == 01
        IF IJ01 == 00
```

```

        SND F0 08 08 J02 2A F7 # Int
    END
    IF IJ01 == 01
        snd F0 08 08 J02 7F F7 # Both
    END
END

# increment looping values
MAT J00 = J00 + 2
MAT J01 = J01 + 2
MAT J02 = J02 + 1
END
END

```

Logic to override effects status when V-piano loaded

Z1 tone is last req in scan.

If V-Piano, override the rcvd Mod/FX & Trem displays

```

    IF M07 == 10 00 20 23 #Z1 Tone one category Address
        IF MB == 41 # tone category 41 is V-piano
            # Override Mod/FX display
            SND F0 41 10 00 00 75 12 10 00 10 01 07 07 51 F7
            # Override tremolo display
            SND F0 41 10 00 00 75 12 10 00 12 01 07 07 4F F7
            # Turn off effect Mod/FX button
            SND F0 41 10 00 00 75 12 10 00 10 00 00 60 F7
            # Turn off tremolo button
            SND F0 41 10 00 00 75 12 10 00 12 00 00 5E F7
        END
    END
END

```

End V-Piano override logic

ModFx / Tremolo Amp Sim Message Parse

```

    Ass TestFlag = True

    IF M07 != 10 00
        Ass TestFlag = False
    END

    IF M09 < 10
        Ass TestFlag = False
    END

```

```
IF M09 > 1E
    Ass TestFlag = False
END
```

```
IF ML != 94
    Ass TestFlag = False
END
```

```
IF TestFlag == True
```

```

    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 00 M0B 00
F7
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 1 M0C M0D
00 F7
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 7 M12 M13
M14 M15 00 +F
    SND F7 +F
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 0B M16 M17
M18 M19 00 +F
    SND F7 +F
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 0F M1A M1B
M1C M1D 00 +F
    SND F7 +F
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 13 M1E M1F
M20 M21 00 +F
    SND F7 +F
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 17 M22 M23
M24 M25 00 +F
    SND F7 +F
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 1B M26 M27
M28 M29 00 +F
    SND F7 +F
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 1F M2A M2B
M2C M2D 00 +F
    SND F7 +F
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 23 M2E M2F
M30 M31 00 +F
    SND F7 +F
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 27 M32 M33
M34 M35 00 +F
    SND F7 +F
    SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 2B M36 M37
M38 M39 00 +F
```

SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 2F M3A M3B
M3C M3D 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 33 M3E M3F
M40 M41 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 37 M42 M43
M44 M45 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 3B M46 M47
M48 M49 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 3F M4A M4B
M4C M4D 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 43 M4E M4F
M50 M51 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 47 M52 M53
M54 M55 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 4B M56 M57
M58 M59 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 4F M5A M5B
M5C M5D 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 53 M5E M5F
M60 M61 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 57 M62 M63
M64 M65 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 5B M66 M67
M68 M69 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 5F M6A M6B
M6C M6D 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 63 M6E M6F
M70 M71 00 +F
SND F7 +F
SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 67 M72 M73

M74 M75 00 +F

SND F7 +F

SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 6B M76 M77

M78 M79 00 +F

SND F7 +F

SND F0 M01 M02 M03 M04 M05 M06 M07 M08 M09 6F M7A M7B

M7C M7D 00 +F

SND F7 +F

END

End ModFx / Trem Parse

END

END

RD 2000 Manager 2.30 StreamByter Outbound

Red Heron Music
March 2025
Additional details in inbound rules

If Load

Alias i0 PgmChl
Alias i1 BankCmd
Alias i2 PcCmd
Alias i3 Temp1
Alias i4 Temp2
Alias i5 Temp3

Alias G0 CurPgm # 1-\$300, parked at 0

These should get overwritten by load triggering a resend
Ass PgmChl = F
Ass BankCmd = BF
Ass PcCmd = CF

End

Handle outbound load name request

1. Use CurPgm to recall the designated program
2. Request

If M0 == F0 58 58 00
If M04 == 00 00
Ass CurPgm = 0
Else
If M04 == 02 0D

Else
Mat Temp1 = M04 * 80
Mat Temp1 = Temp1 + M05

Pull out current program # 1-300
Ass CurPgm = Temp1

```
# Calculate bank & program change values
# MSB = 84, LSB = 64-66, PC = 0-1127
```

```
Mat Temp1 = Temp1 - 1
```

```
Mat Temp2 = Temp1 / 80
Mat Temp2 = Temp2 + $64
Mat Temp3 = Temp1 % 80
```

```
Snd BankCmd 00 $84
Snd BankCmd 20 Temp2
Snd PcCmd Temp3
```

```
# Request program name, with slight delay
```

```
Snd F0 41 10 00 00 75 11 +F +D100
Snd 10 00 00 00 00 00 00 10 00 F7 +F +D100
```

```
End
```

```
End
Block
```

```
End
```

Handle outbound Program Control Channel

```
If M0 == F0 58 58 04
  Ass PgmChl = M04
  Mat BankCmd = B0 + PgmChl
  Mat PcCmd = C0 + PgmChl
  Block
End
```