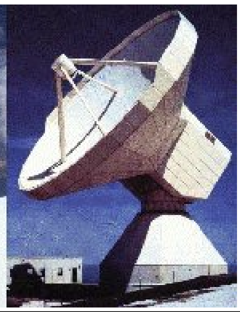



IRAM-NNNNN

Revision: 0
2004-11-19

Contact Author

Institut de RadioAstronomie Millimétrique

Plateau de Bure: CAN control of the 22 GHz Receiver and of the Antenna Subreflector

Owner Francis Morel (morel@iram.fr)

Keywords:

Approved by:	Date:	Signature:
A.Perrigouard		

Change Record

REVISION	DATE	AUTHOR	SECTION/PAGE AFFECTED	REMARKS
1	July 8 2004	Francis Morel		
2	Nov 4 2004	Francis Morel	Added SUBREF	

Contents

1.

1 DESCRIPTION OF THE "CAN2VME" INTERFACE

This interface was designed to allow retrofitting VME boards designed by IRAM, mainly the "22G" board, used for the 22GHz radiometer control, and the "SUBREF" board, in charge of the antennas' subreflector control for homology correction The goal of the interface is to drive these VME boards under control of a CAN bus. The CAN controller is a C164 Microprocessor with built-in Full-CAN interface. The CAN controller acts as a CAN to VME bridge, emulating the VME Bus protocol.

1.1 CAN2VME Controller functionalities:

The controller is **NOT** able to drive a VME crate, for several reasons:
-It does not fulfill the requirements for all addressing modes used by VME. Only Address bits 1 to 8 are driven by the controller, the Address Modifier is fixed to 0x29 (most common case: A16, D16).
-It has no connection to the P2 connector of the VME backplane, and is thus unable to make 32-bit access.
-It does not handle all interrupt levels; the controller only monitors IRQ4.
But, as it is, this controller is compatible with most standard 16-bit VME boards.

2 HARDWARE:

2.1 CAN2VME Controller description:

The board is built around a DIP-164 module developed by Systec. This module is a 40-pin DIP component, it includes Microprocessor, 32k RAM, 32k Flash-EPROM, RS-232 interface, CAN interface. The CAN is not opto-isolated. The VME is interfaced to this board through a 8-bit I/O bus and 8 control signals. The VME time-out was fixed to 64 usec. 2 push buttons on the board are used for **"Reset"** and **"Boot",** and a HE-10 10 pos connector is available for **RS-232** connection. These are reserved functions, and none of them is accessible when the controller board is inserted into the chassis. See schematics for more details.

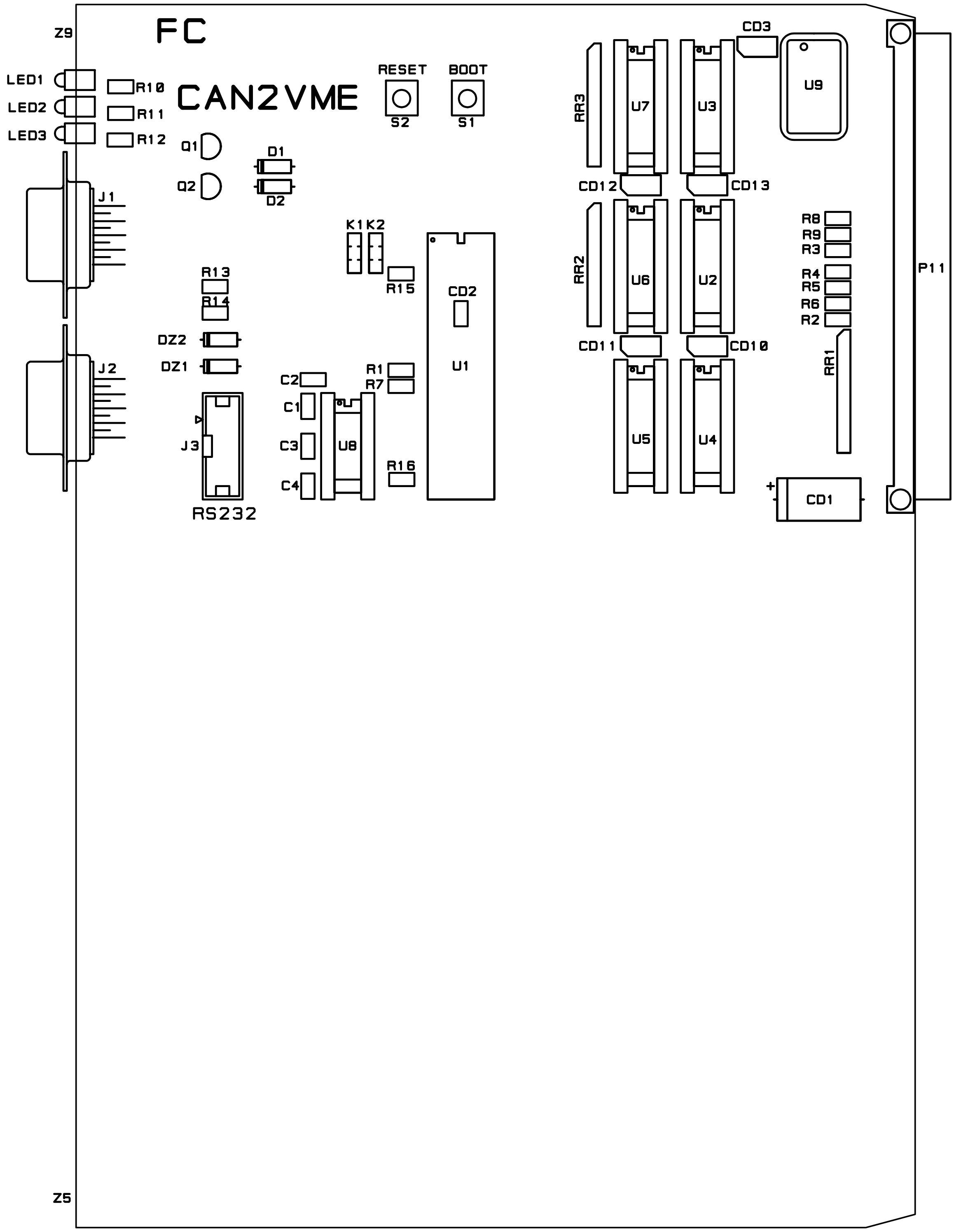
2.1.1 Controller chassis:

The controller is enclosed in a small (6U, 2-slot) tabletop chassis. The chassis is powered (5V, 3A). Slot 1 is assigned to the CAN2VME, slot 2 to the SUBREF board, and slot 3 to the 22G VME board.

2.1.2 Controller front-panel:

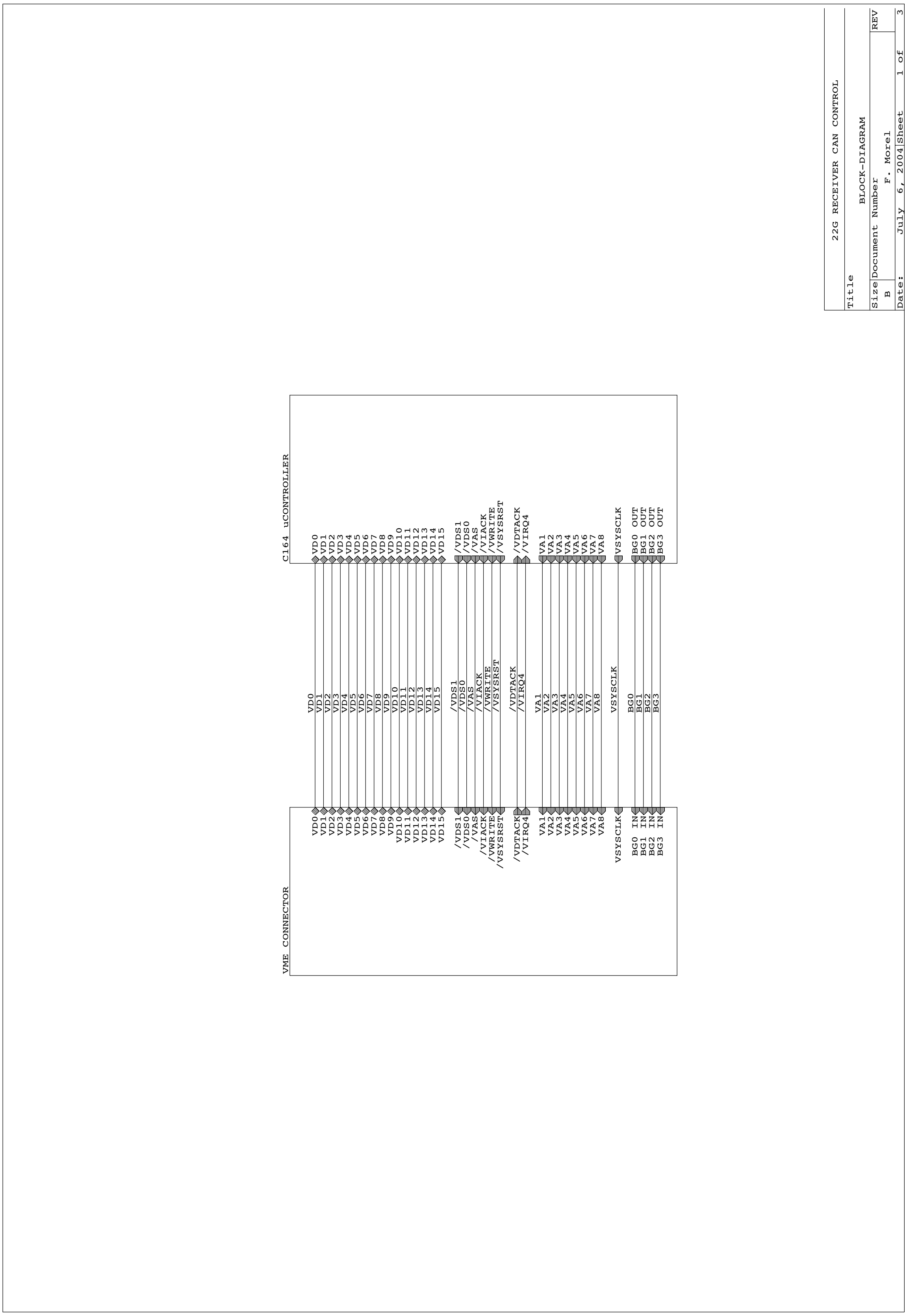
3 LEDS monitor :
-**Power** (green)
-**CAN Message**(yellow)
-**CAN Error** (red**).**
2 DB-9 male connectors are used for connection to the CAN bus. The 2 connectors are tied pin-to-pin.
Connections:
2: CAN Low
7: CAN High
3: CAN Gnd

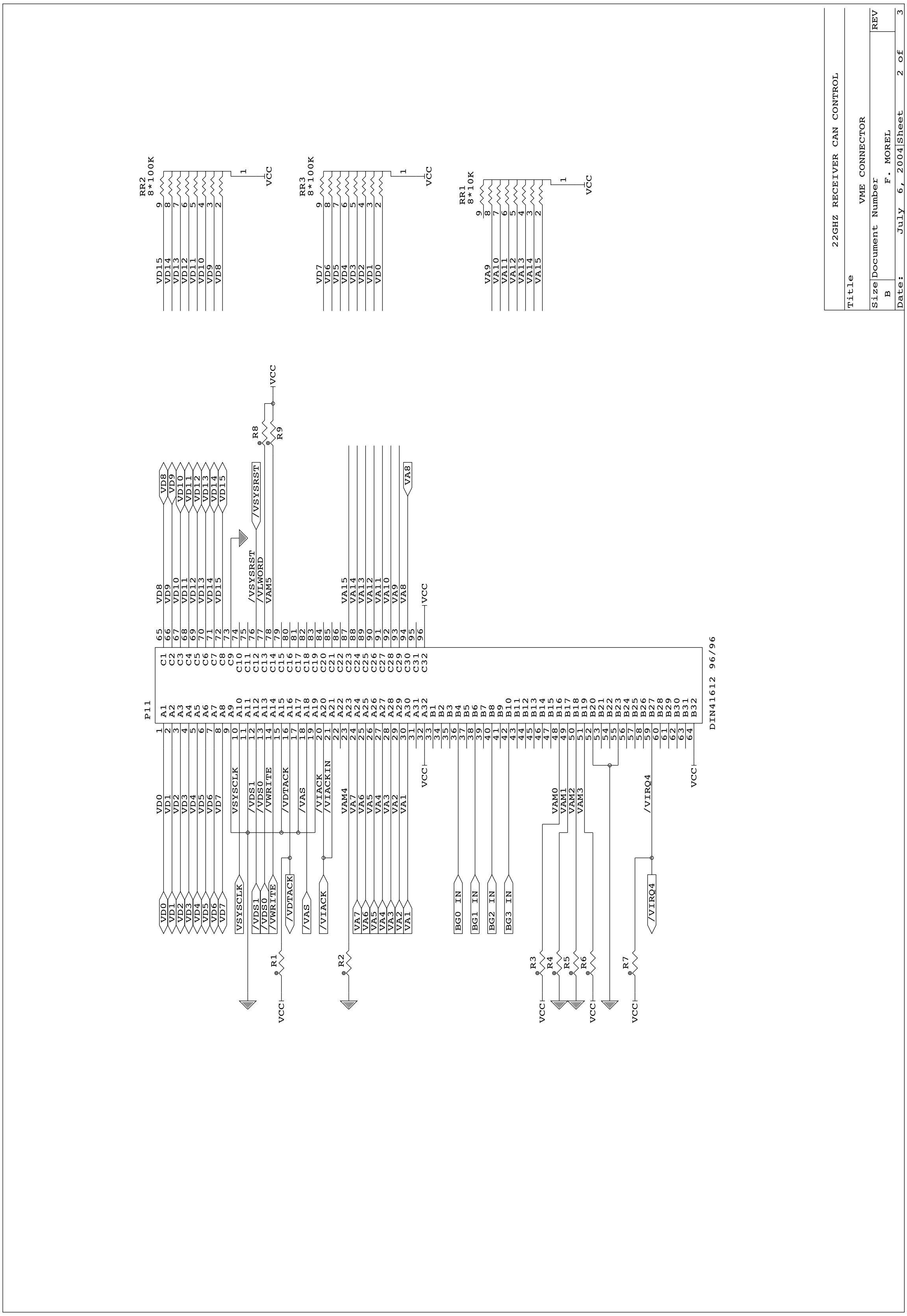
2.1.3 Controller layout:

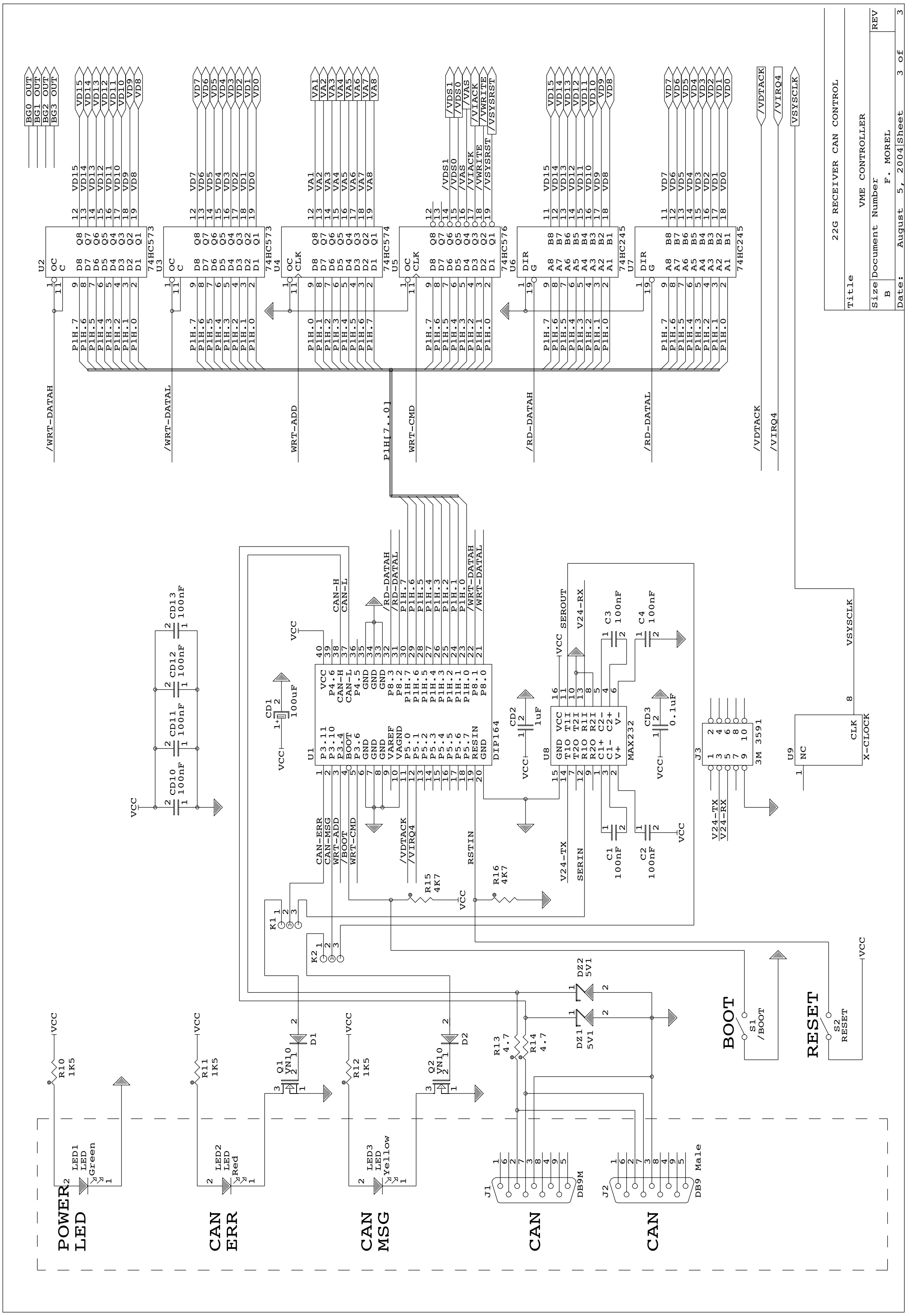


For compatibility with the VME mechanical standard, the controller is a 6U board, but it may simply be cut to 3U if necessary.

2.1.4 Controller schematics




22GHZ RECEIVER CAN CONTROL
Title
VME CONNECTOR
Size Document Number
B F. MOREL
REV
Date: July 6, 2004 Sheet 2 of 3
DIN41612 96/96
P11
RR1 8*10K
RR2 8*100K
RR3 8*100K


22G RECEIVER CAN CONTROL
VME CONTROLLER
F. MOREL
August 5, 2004
Sheet 3 of 3
POWER LED
CAN ERR
CAN MSG
CAN
CAN
BOOT
RESET

2.2 22G VME Board description:

This board was specifically designed to control the 22G Receiver. It connects to the Interferometer “TU01” (1 Hz) pulse, fed to all antennas from the GPS time base, through a front-panel BNC connector. This input signal is used to (re)synchronize a local oscillator. It connects to the Receiver through a bi-directional optical link, to avoid interferences and ground loops.
The board generates an interrupt and latches the status and the data from the Receiver upon each valid "TU01" pulse. The vector used by the interrupt depends on the board status: If the board is unlocked (not synchronized with TU01), the vector ERROR will be used. Otherwise, vector OK.
The main parts of this board are 6 [31-bit + overflow] counters, used to integrate the V-to-F outputs of the Receiver. These counters are first registered and then cleared upon each "TU01" pulse. The blanking (lost each second) time is typically 180 nanoseconds.
The board was built using a FPGA Altera EPF10K30.
The VME BUS is used as a 16-bit wide bus (D16 norm).This implies that each readout of a 32-bit word will need 2 accesses on the VME Bus.
The interface between this board and the receiver uses 12 low-speed, low-cost plastic fiber optics :

2.2.1 Fiber outputs TO the receiver :

CMD_ LOAD_ON	Lit fiber moves the reference load in front of the receiver.	Static command bit
CMD_NOISE_ON	Lit fiber turns ON the noise diode of the receiver.	Static command bit
CMD_PWR	Unused, but functional	Static command bit

2.2.2 Fiber inputs FROM the Receiver:

LOAD_ON	Fiber is lit if the Reference Load is in front of the Receiver	Static Status bit
2 MHz Reference	V/F reference frequency	Frequency encoded signal
LOAD_T	Reference Load temperature	Frequency encoded signal
PELTIER_T	Peltier regulator temperature	Frequency encoded signal
ALARM	Receiver alarm	Static Status bit
F3	Channel 3	Frequency encoded signal
F2	Channel 2	Frequency encoded signal
F1	Channel 1	Frequency encoded signal
F0	Channel 0	Frequency encoded signal

2.2.3 22G registers addresses:

. All the registers are mapped in the A16/D16 VME I/O space => Address Modifier=29/2D.The board uses 128 word (even) addresses (XX00 to XXFE). The board Base-address bits [15..8] are selectable using 2 encoding wheels, **RC1 (A15..A12)**, and **RC2 (A11..A08).**
The actual VME address on Plateau de Bure of the Subref Boar d is 0xFFFFFF00

2.2.4 Input registers (read-only):

Channel 0 LSW	Base-address + 0x0	Channel 0 [15..0]
Channel 0 MSW	Base-address + 0x2	Bit 15 = Overflow Bits [14..0]=Channel 0 [30..16]
Channel 1 LSW	Base-address + 0x4	Channel 1 [15…0]

Channel 1 MSW	Base-address + 0x6	Bit 15 = Overflow Bits [14..0]=Channel 1 [30..16]
Channel 2 LSW	Base-address + 0x8	Channel 2 [15…0]
Channel 2 MSW	Base-address + 0xA	Bit 15 = Overflow Bits [14..0]=Channel 2 [30..16]
PELTIER_T LSW	Base-address + 0xC	Peltier temp[15...0]
PELTIER_T MSW	Base-address + 0xE	Bit 15 = Overflow Bits [14..0]=Load temp [30..16]
LOAD_T LSW	Base-address + 0x10	Peltier temp [15...0]
LOAD_T MSW	Base-address + 0x12	Bit 15 = Overflow Bits [14..0]=Load temp [30..16]
2 MHZ LSW	Base-address + 0x14	2 MHz Ref [15..0]
2 MHZ MSW	Base-address + 0x16	Bit 15 = Overflow Bits [14..0]=2MHz Ref [30..16]
Channel 3 LSW	Base-address + 0x18	Channel 3 [15…0]
Channel 3 MSW	Base-address + 0x1A	Bit 15 = Overflow Bits [14..0]=Channel 3 [30..16]

2.2.5 Vectors register (read-only):

Vector OK LSByte	Base-address + 0x1C	Bits [3..0] = Vector OK [3..0]
Vector ERROR LSByte	Base-address + 0x1C	Bits [11..8]=Vector ERROR [3..0]

2.2.6 Status register (read-only): Base-address + 0x1E

N.B : Status register bits are latched upon “TU01” interrupt edge.

15	14..6	5	4	3	2	1	0
ERR	XX	ALARM	UNL	IT_ENA	NOISE_ON	LOAD_ON	XX

Description of the STS register bits :

Bit 15	ERR	ERR = ALARM + UNL
Bits [14..6]	XX	Don’t care
Bit 5	ALARM	Set if Receiver Alarm is ON
Bit 4	UNL	Set when the board is not synchronized with “TU01”
Bit 3	IT_ENA	Set when the interrupt has been enabled (see Command Register)
Bit 2	NOISE_ON	Set when the Noise Diode has been requested to turn ON. This bit is **NOT** read from the receiver
Bit 1	LOAD_ON	Set when the Reference Load is in front of the Receiver
Bit 0	XX	Don’t care

2.2.7 Vectors registers (write-only):

Vector OK	Base-address + 0x1A	Bits [3..0] = Vector OK [3..0] Vector OK [7..4] are switch-selectable (S1)
Vector ERROR	Base-address + 0x1C	Bits [3..0] = Vector ERROR [3..0] Vector ERROR [7..4] are switch-selectable (S1)

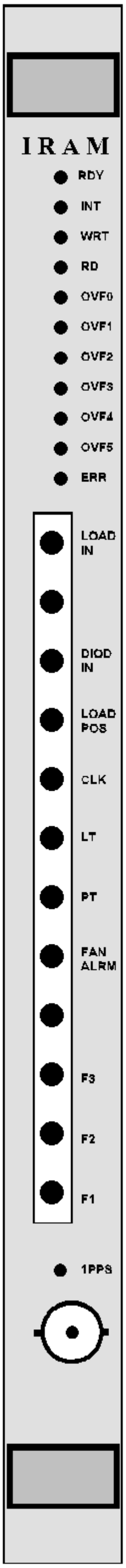
2.2.8 Command Register (write-only): Base-address + 0x1E

15..4	3	2	1	0
XX	CMD_IT_ENA	CMD_NOISE_ON	CMD_LOAD_ON	CMD_PWR

Description of the CMR bits :

Bits [15..4]	XX	Don’t care
Bit 3	CMD_IT_ENA	When set, enables interrupt generation. Cannot be set if both vectors (OK and ERROR) have not been defined first.
Bit 2	CMD_NOISE_ON	When set, turns On the Noise Diode.
Bit 1	CMD_LOAD_ON	When set, moves the Load in front of the Receiver.
Bit 0	CMD_PWR	Unused, but functional

2.2.9 22G VME Board Front-Panel:



Red LEDs are used for displaying ERROR conditions only.

From top to bottom:

11 LEDs :

READY	Green	The interrupt is enabled AND the board is synchronized
INT	Yellow	ON when an interrupt request is active
WRT	Yellow	Will flash upon each “write” access
READ	Yellow	Will flash upon each “read” access
OVF0	Red	Channel 0 Overflow
OVF1	Red	Channel 1 Overflow
OVF2	Red	Channel 2 Overflow
OVF3	Red	Channel 3 Overflow
OVF4	Red	Peltier_T Overflow
OVF5	Red	Load_T Overflow
ERR	Red	The Altera chip did not initialize upon power-on

N.B: All leds "OVFx" will blink if ALARM = 1.

3 optical outputs (see Command register)

Output name	**Signal name**	**Comments**
LOAD IN	CMD_LOAD_ON	When ON, move the reference load in front of the receiver
	CMD_PWR	Unused in actual design, but functional
DIOD IN	CMD_NOISE_ON	When ON, turns ON the noise diode of the receiver

9 optical inputs (see Status register)

Input name	**Signal name**	**Comments**
LOAD POS	LOAD_ON	Turns ON when the reference load is in front of the receiver
CLK	2 MHZ	2 MHz reference from the receiver
LT	LOAD_T	Reference Load temperature measurement
PT	PELTIER_T	Peltier cooler temperature measurement
FAN ALRM	ALARM	Receiver alarm
	F3	Channel 3 measurement
F3	F2	Channel 2 measurement
F2	F1	Channel 1 measurement
F1	F0	Channel 0 measurement

1 yellow LED:
"pps" pulse, "TU01" resynchronized and regenerated by the board.

1 BNC connector:
"TU01" input.

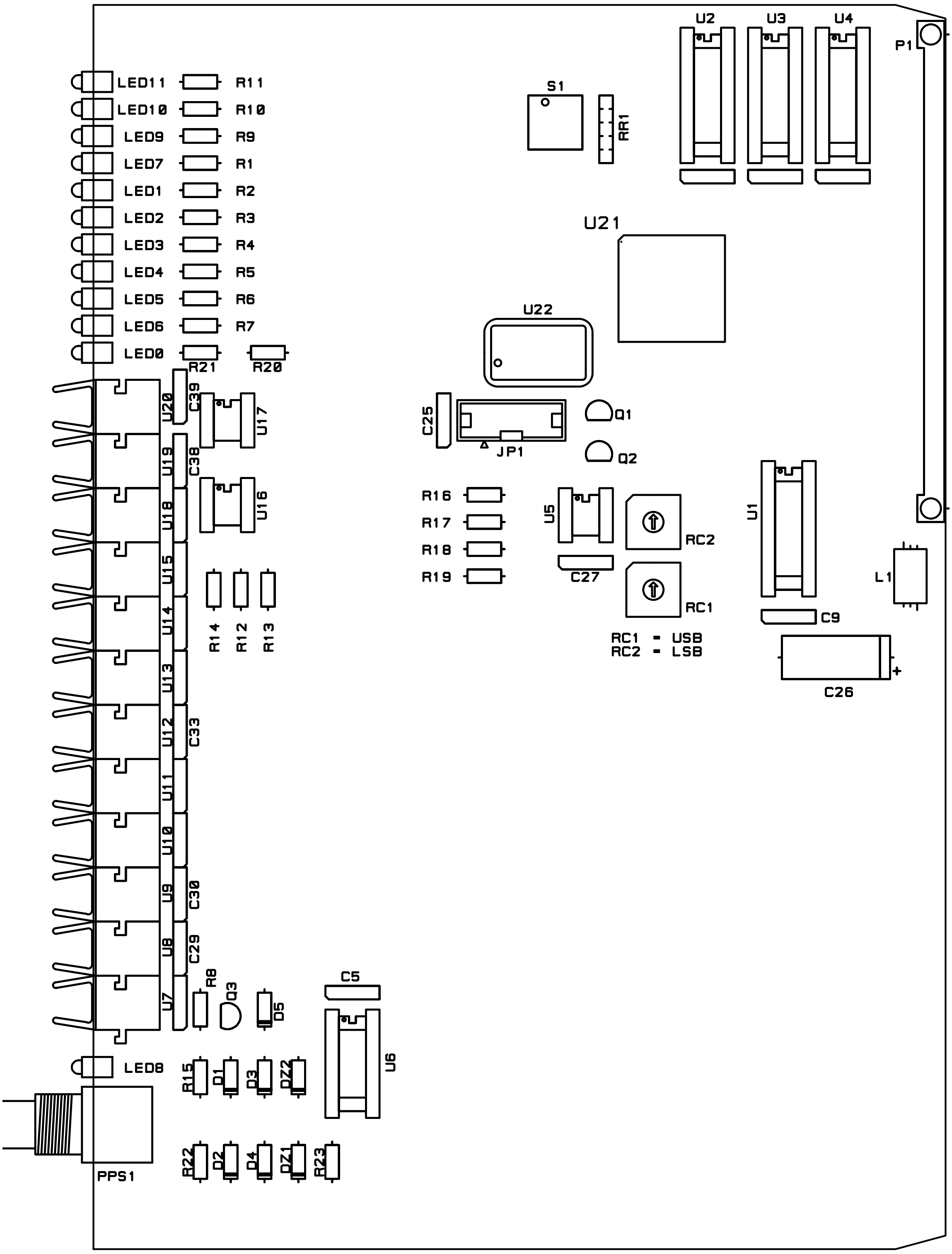
2.2.10 Synchronization:

Upon start-up, the board is in "start" mode and generates no interrupt.
It accepts the first "TU01" pulse it receives, starts its internal base-time, and waits for the next pulse. If this next pulse is received within 1 second +/- 4 milliseconds, the board goes into "synchronized" mode, accepting only the pulses separated by 1+/- 4e-3 seconds, resynchronizing on each of these pulses, and generating each time an interrupt (if the bit CMD_IT_ENA has been set in the Command register).
Glitches are thus eliminated.
If the "TU01" pulse is missing, the board will then internally supply this pulse and still generate the interrupt, but no more than 32 times. After a delay of 32 seconds without having received any "TU01" pulse, the board will go back into "start" mode and stop generating interrupts and latching data.

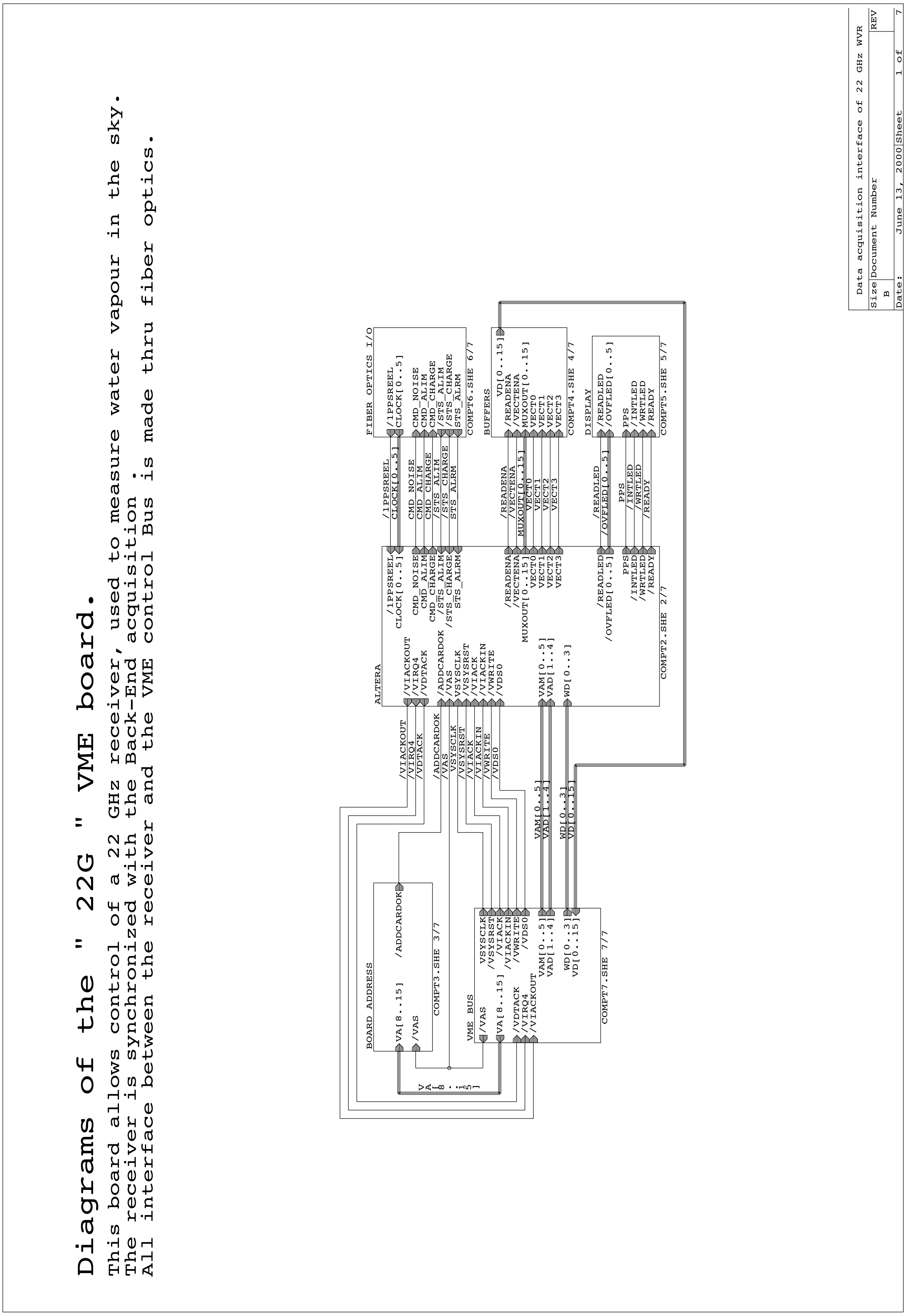
2.2.11 Getting started:

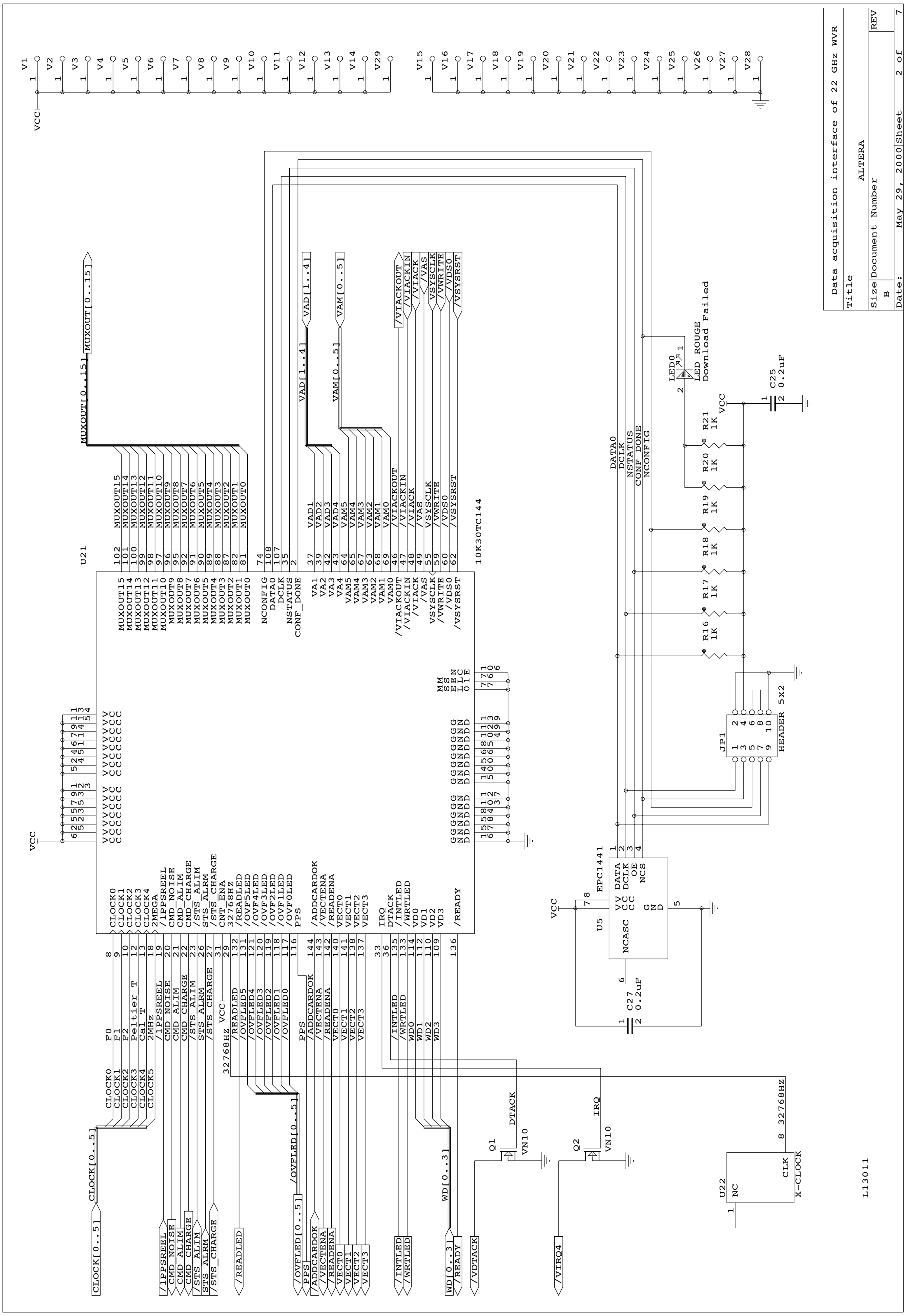
Select Base-Address [A15...A8] using RC1 and RC2.
Select High Nibble [0..F] of the vectors using S1.
Insert the board into the VME crate.
Connect the Receiver’s fiber optics
Connect the "TU01" input connector.
Turn on the crate. The yellow "pps" LED should flash at 1 Hz. All red OVF leds also blink if the Status bit ALARM is ON.
Using the debugger:
-write vector OK (Base-Address + 0x1A) Low Nibble (0...F).
-write vector ERROR (Base-Address + 0x1C) Low Nibble (0...F).
-Enable interrupts, writing data "0x8" at (Base-Address + 0x1E): The green LED "READY" should turn ON.

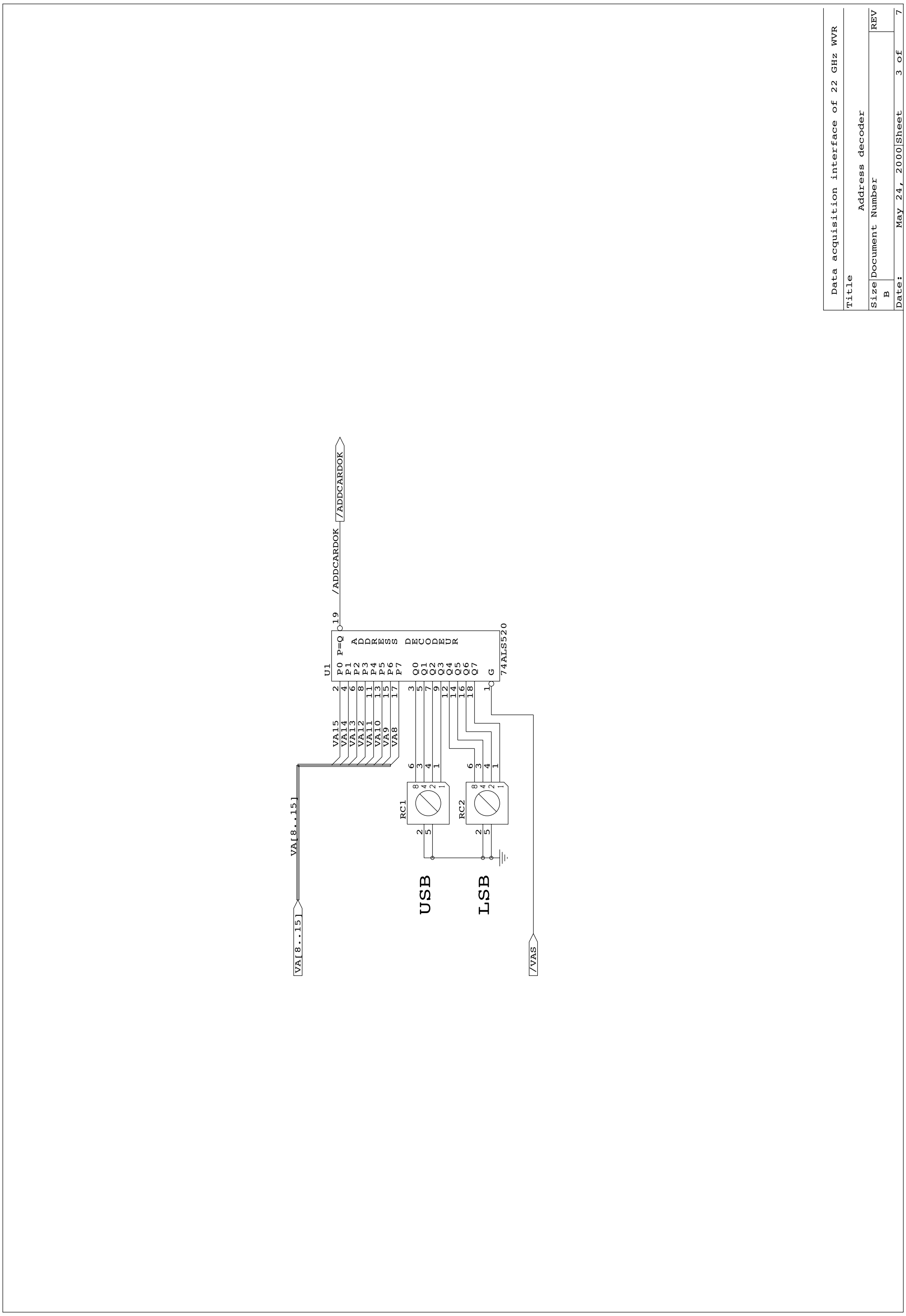
2.2.12 22G VME Board layout:

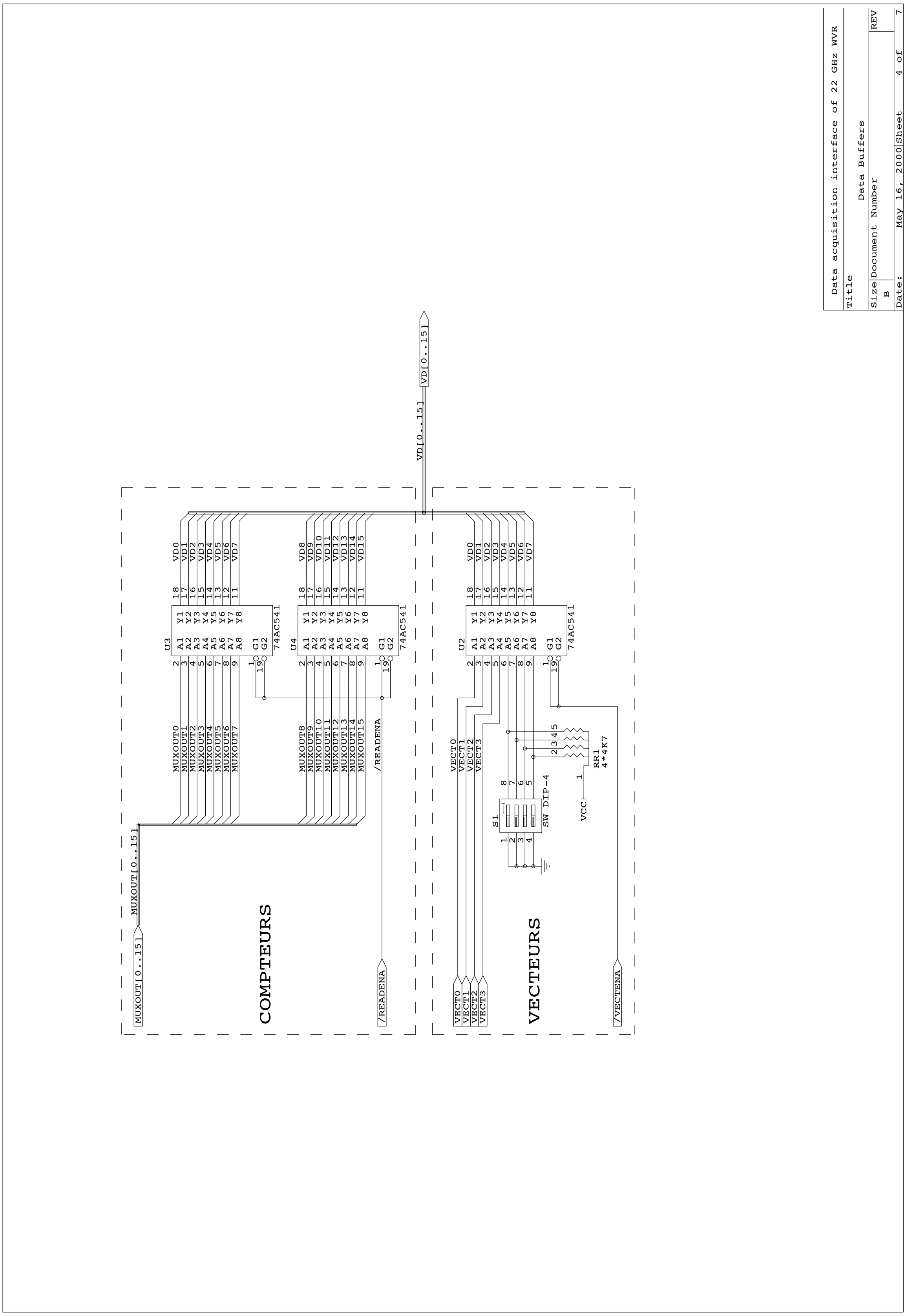


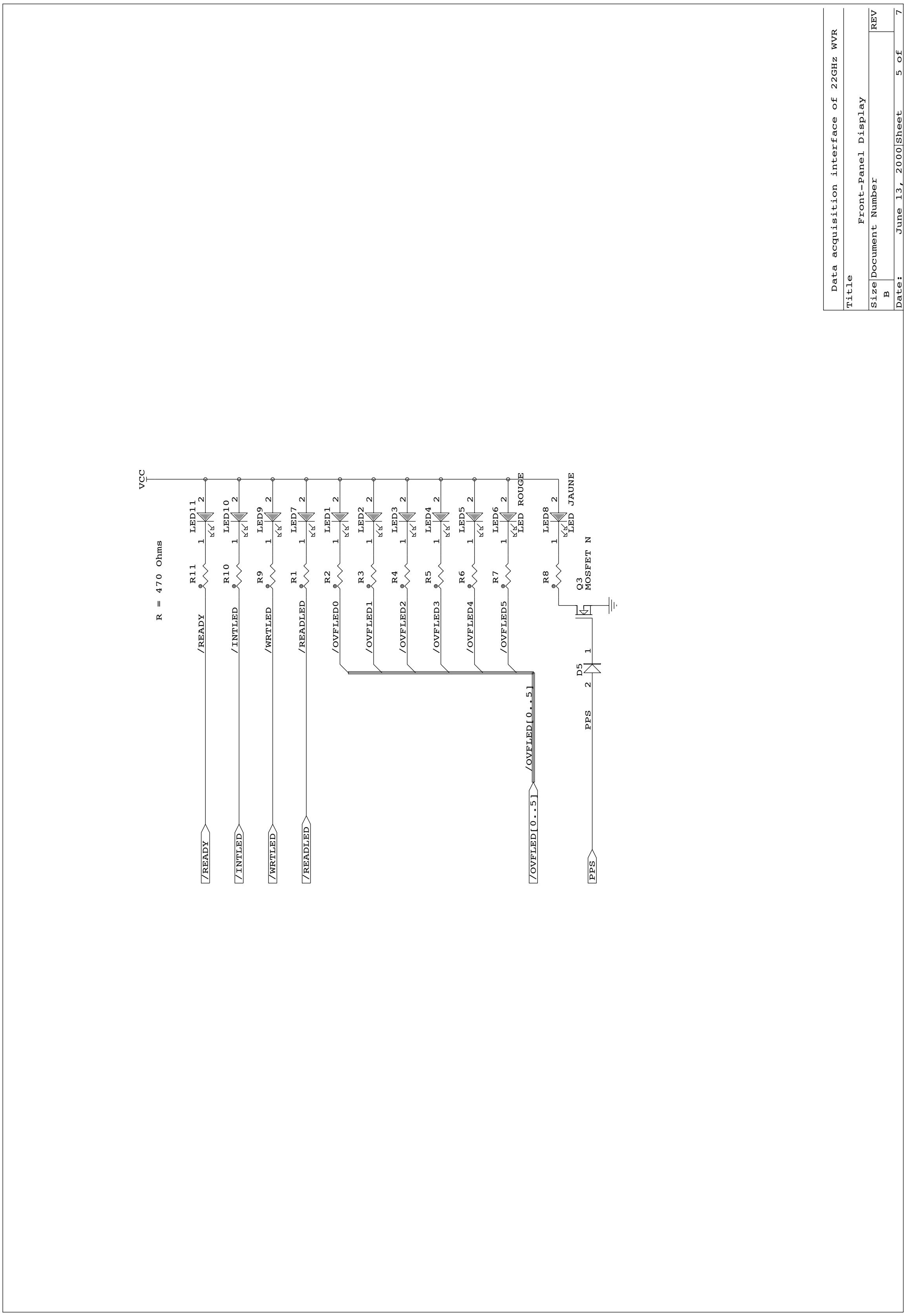
2.2.13 22G VME Board schematics:

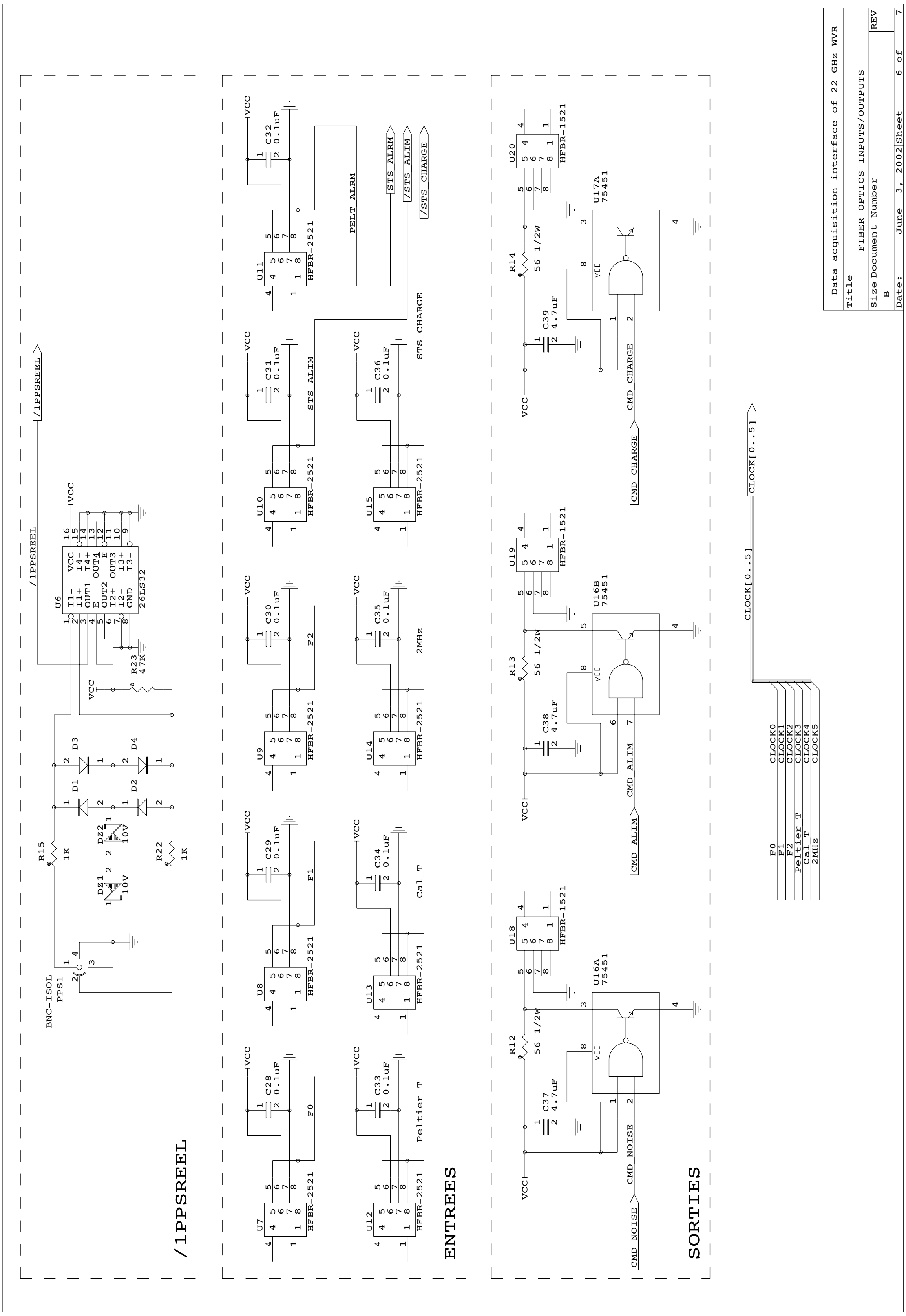


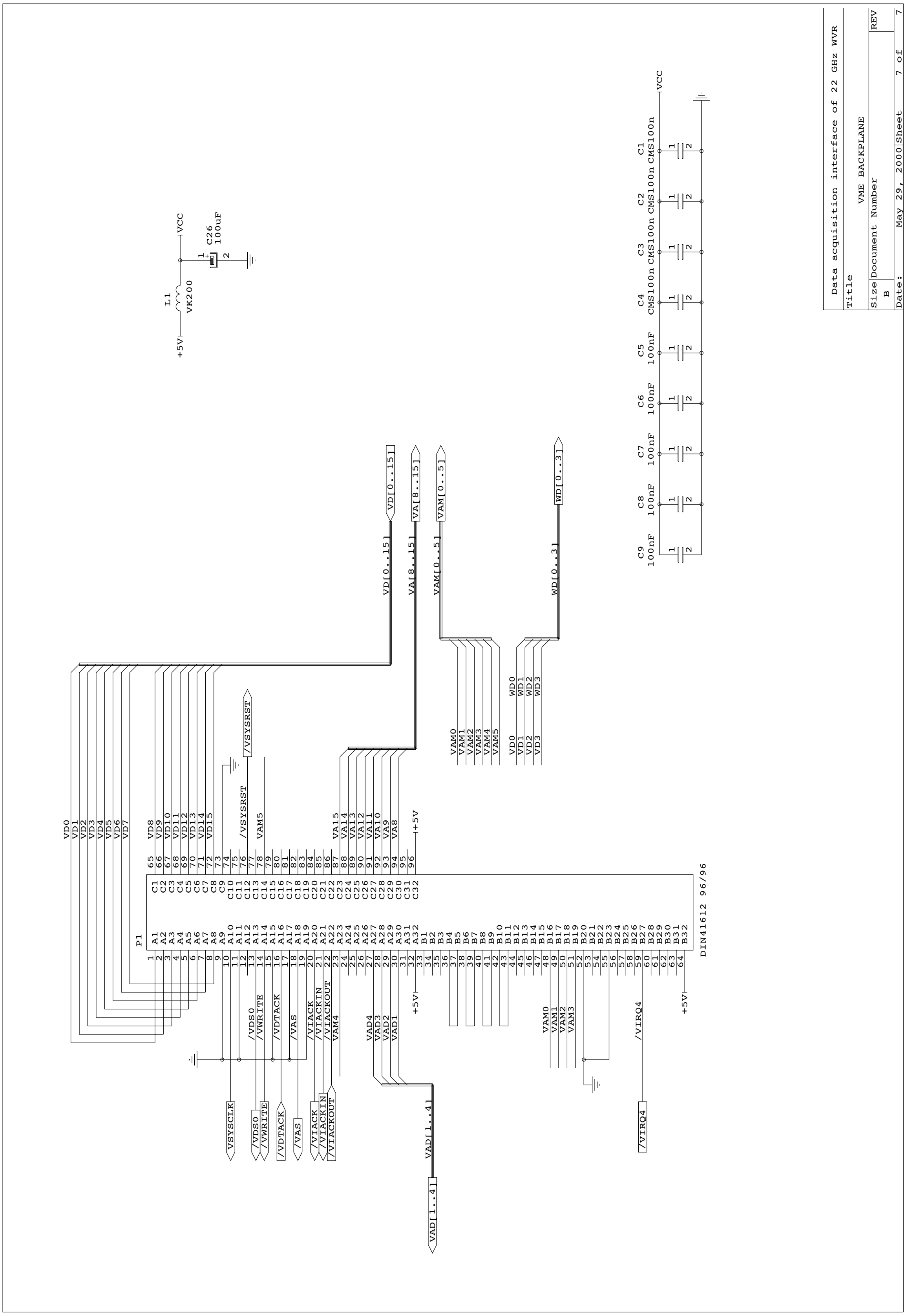

Data acquisition interface of 22 GHz WVR
Title
ALTERA
Size Document Number
REV
B
Date: May 29, 2000 Sheet 2 of 7
U21
10K30TC144
U5
EPC1441
U22
L13011
JP1
HEADER 5X2
LED0
LED ROUGE
Download Failed
C25 0.2uF
C27 0.2uF
Q1 VN10
Q2 VN10


Data acquisition interface of 22 GHz WVR
Title
Address decoder
Size
B
Document Number
REV
Date: May 24, 2000
Sheet 3 of 7
/ADDCARDOK
U1
74ALS520
ADDRESS DECODEUR
P=Q
VA[8..15]
RC1
RC2
USB
LSB
/VAS


Data acquisition interface of 22 GHz WVR
Title
Data Buffers
Size Document Number REV
B
Date: May 16, 2000 Sheet 4 of 7
VD[0..15]
COMPTEURS
MUXOUT[0..15]
/READENA
U3
U4
74AC541
VECTEURS
U2
VECT0
VECT1
VECT2
VECT3
/VECTENA
S1
SW DIP-4
RR1
4*4K7
VCC


Data acquisition interface of 22GHz WVR
Title
Front-Panel Display
Size Document Number
B
REV
Date: June 13, 2000 Sheet 5 of 7
VCC
R = 470 Ohms
LED11 LED10 LED9 LED7 LED1 LED2 LED3 LED4 LED5 LED6
LED ROUGE
LED8
LED JAUNE
R11 R10 R9 R1 R2 R3 R4 R5 R6 R7 R8
Q3
MOSFET N
D5
/READY
/INTLED
/WRTLED
/READLED
/OVFLED0
/OVFLED1
/OVFLED2
/OVFLED3
/OVFLED4
/OVFLED5
/OVFLED[0..5]
PPS


Data acquisition interface of 22 GHz WVR
Title
FIBER OPTICS INPUTS/OUTPUTS
Size
Document Number
REV
B
Date: June 3, 2002
Sheet 6 of 7
/1PPSREEL
ENTREES
SORTIES
CLOCK[0..5]
CMD NOISE
CMD ALIM
CMD CHARGE
STS ALRM
/STS ALIM
/STS CHARGE


Data acquisition interface of 22 GHz WVR
Title
VME BACKPLANE
Size Document Number REV
B
Date: May 29, 2000 Sheet 7 of 7

2.3 SUBREF VME Board description

This board is a slave device, under control of a remote microprocessor.
This board was designed for Subreflector control. The subreflector hyperbolic mirror needs dynamic correction to compensate for the main mirror deformation under gravity, depending on the antenna elevation. Tilt, translations and focus adjustments are handled.
The Subref board drives 5 motors, moving the mirror as requested by the microprocessor. The Subref board is composed of a carrier board in charge of the VME Bus interface, and 5 similar daughter "Submot" boards, each "Submot" driving one motor.
The motors are powered and connected to the Submot boards through an electronic rack, which also provides an optical isolation. This is the rack "SubIsol", which is not described in this documentation.

2.3.1 Motors and associated logics:

The 5 motors (mot1 to mot5) are similar. Each of them is a DC motor equipped with an encoder delivering 64 periods of a Sine/Cosine TTL signal per motor revolution. The encoder does not supply any reference pulse. The impulsions of the encoder are subdivided by 64 in the electronics, which allows counting the motor revolutions with a resolution of 1 revolution.
Each motor is equipped with a precision microswitch. A bit of the Status register (SWI[x]) reflects the state of this switch: SWI[x] = 0 when motor[x] is positioned in the normal displacement zone, and SWI[x] = 1 when motor[x] has reached its stroke limit in negative direction. The switch is also used as a hardware limit switch and SWI[x] = 1 will forbid motor[x] moves with negative velocity.
The Subref board allows reading the actual position (16-bit signed APOS signed registers) and setting the requested position (16-bit RPOS signed registers) and requested velocity of each motor. Setting a velocity request bit (PVR[x] or NVR[x] of the CMR, x being the motor number [1..5]) forces the selected motor to move with requested constant velocity. A move towards the switch has negative velocity. A velocity request is always effective.
For position control, the motor position has to be initialized first. This is done through the 16-bit Command register (CMR) and checked through the 16-bit Status register (STS) of the Subref board.

Command Register (CMR):

15	14	13	12	11	10	09	08	07	060	05	04	03	02	01	00
TST	NVR5	PVR5	ENA5	NVR4	PVR4	ENA4	NVR3	PVR3	ENA3	NVR2	PVR2	ENA2	NVR1	PVR1	ENA1

TST: May be set/reset (for tests only).
NVR[x]: forces motor[x] to move with constant negative velocity.
PVR[x]: forces motor[x] to move with constant positive velocity.
ENA[x]: If ID[x] = 0, ENA[x] preloads motor[x] position register to zero, upon transition (from 0 to 1) of SWI[x]. Bit ID[x] of Status is then set. Resetting ENA[x] resets ID[x].

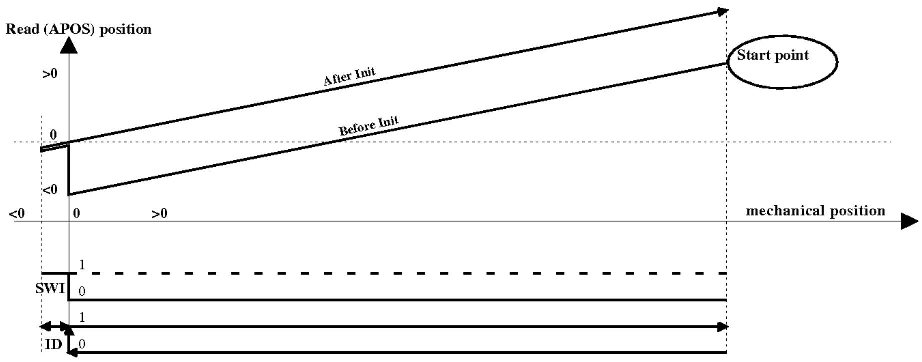
Status Register (STS):

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
TST	RUN5	ID5	SWI5	RUN4	ID4	SWI4	RUN3	ID3	SWI3	RUN2	ID2	SWI2	RUN1	ID1	SWI1

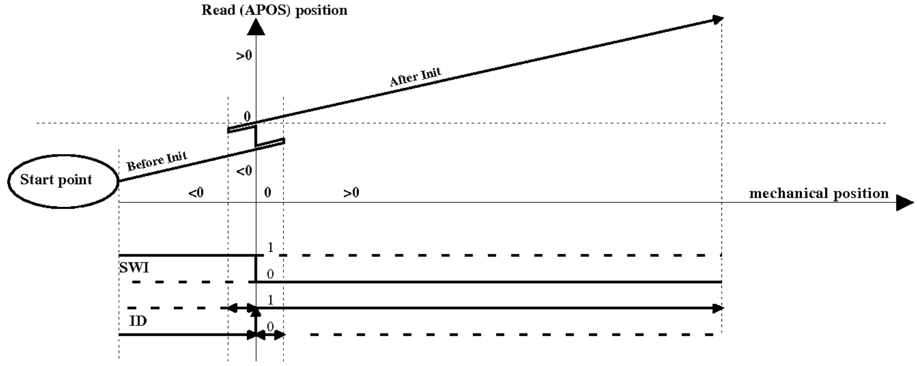
TST: recopy of bit TST of the CMR.
RUN[x]: motor[x] is requested to move, in any way.
ID[x]: Init Done[x] = 1 when motor[x] has been initialized.
SWI[x]: Switch of motor[x], set if motor[x] has reached its negative limit.

2.3.2 Motors' init:

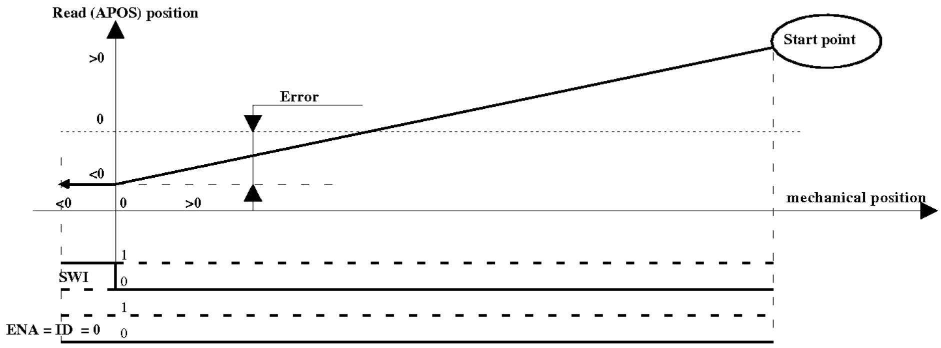
Upon turn-on, the actual position of a motor is unknown, as it is not initialized. For initialization, following commands are used:
-SWI[x] is checked first:
If SWI[x] = 1, PVR[x] is set, forcing motor[x] to run at constant velocity in the positive direction, until SWI[x] = 0. The motor stops, PVR[x] is then reset.
-If SWI = 0, NVR[x] and ENA[x] are set, forcing motor[x] to move with constant negative velocity until SWI[x] = 1. Upon SWI[x] transition (from 0 to 1), the board preloads APOS[x] register to 0x0000 and sets ID[x]. The motor stops, it is now initialized.
-Resetting NVR[x] will set motor[x] in position control mode.



SUBREF: Motor position Init (starting with SWI = 0)



SUBREF: Motor position Init (starting with SWI = 1)



SUBREF: Checking motor position reliability (Error should be zero)

2.3.3 Position control mode:

Once in this mode, motor[x] will go to the requested position. This position value is writable in the Requested Position Register, RPOS[x]. The actual position is readable in the Actual Position Register (APOS[x]). As might be expected, a move with negative velocity (move down) will decrease the position value. Positive velocity (move up) move will increase it.

2.3.4 Caveat user:

-Upon turn-on, all bits of the CMR are reset. Thus, ID[x] of the Status are reset. All bits of APOS[x] and RPOS[x] registers are also reset.
-A VME “Reset” has **NO** effect on the board.
-Bits PVR[x] and NVR[x] are mutually exclusive and if both are set, motor[x] always stops.
-Setting PVR[x] = 1 and NVR[x] = 0 causes motor[x] to move at constant positive velocity.
-Setting PVR[x] = 0 and NVR[x] = 1 causes motor[x] to move at constant negative velocity.
-When NVR[x] = 0 AND PVR[x] = 0 AND **ID[x] = 1,** motor[x] will always go to RPOS[x] position.
-If ENA[x] = 0, SWI[x] transition (from 0 to 1) will NOT preload APOS[x] to zero, but will “freeze” the contents of APOS[x]. This sampling is used for checking the reproducibility of position “zero”.

2.3.5 VME interface:

The board is a A16/D16 standard VME board. It is mapped in the VME 16-bit “short” address space.
All register addresses are relative to the Base Address selected with the encoding wheels **RC2(A15-A12)** and **RC1(A11-A8)**.
-The actual address on Plateau de Bure of the Subref Board is 0xFFFFFE00.

Register name	Register address	Data
STS	0x00	Status register (Read only)
APOS1	0x04	Actual Position of Motor1 (Read only)
APOS2	0x08	Actual Position of Motor2 (Read only)
APOS3	0x0C	Actual Position of Motor3 (Read only)
APOS4	0x10	Actual Position of Motor4 (Read only)
APOS5	0x14	Actual Position of Motor5 (Read only)
CMR	0x00	Command Register (Write only)
RPOS1	0x04	Requested Position of Motor1 (Write only)
RPOS2	0x08	Requested Position of Motor2 (Write only)
RPOS3	0x0C	Requested Position of Motor3 (Write only)
RPOS4	0x10	Requested Position of Motor4 (Write only)
RPOS5	0x14	Requested Position of Motor5 (Write only)

2.3.6 Switches, jumpers:

-Subref Base Address (address bits [A15…A8]) is selectable using 2 rotary encoding wheels on the board.

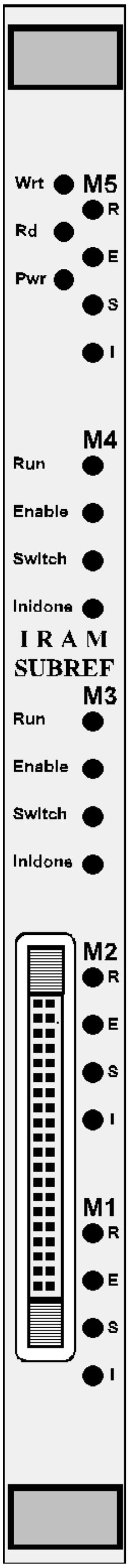
Encoding wheel	VME address bits
RC2 (0 to F)	A15..A12
RC1 (0 to F)	A11..A8

-Each Submot board samples the encoders’ signals; it detects, filters and counts the rotations. The sample frequency is selectable with **jumper SK1 on a 16-pin DIP socket**:

Jumper SK1 position	Sampling frequency
1-16	4 MHz
2-15	2 MHz
3-14	1 MHz
4-13	500 kHz
5-12	**250 kHz (default)**
6-11	125 kHz
7-10	62.5 kHz
8-9	31.25 kHz

This adjustment depends on the complete (motor + encoder + logics) configuration, and should not be modified.

2.3.7 SUBREF VME Board Front-Panel:



2.3.8 Front-panel display:

The main carrier board, as well as the Submot boards uses FPGA chips. This allows easy modification of the functions. This also guarantees the short response times necessary for real-time motors driving.

The carrier board is in charge of the VME transactions. It drives 3 LEDs, displaying VME access and power status.

Each Submot drives 4 LEDs: Visible on the front-panel as 5 groups of 4 LEDs , they allow knowing at a glance the status of each motor.

LED name	Led colour	LED function
Wrt	Yellow	VME Write access display
Rd	Yellow	VME Read access display
Pwr	Green	VME Power On display
Run (Mx)	Yellow	RUN[x] display
Enable(Mx)	Green	ENA[x] display
Switch(Mx)	Red	SWI[x] display
Inidone(Mx)	Green	ID[x] display

2.3.9 Front-panel connector:

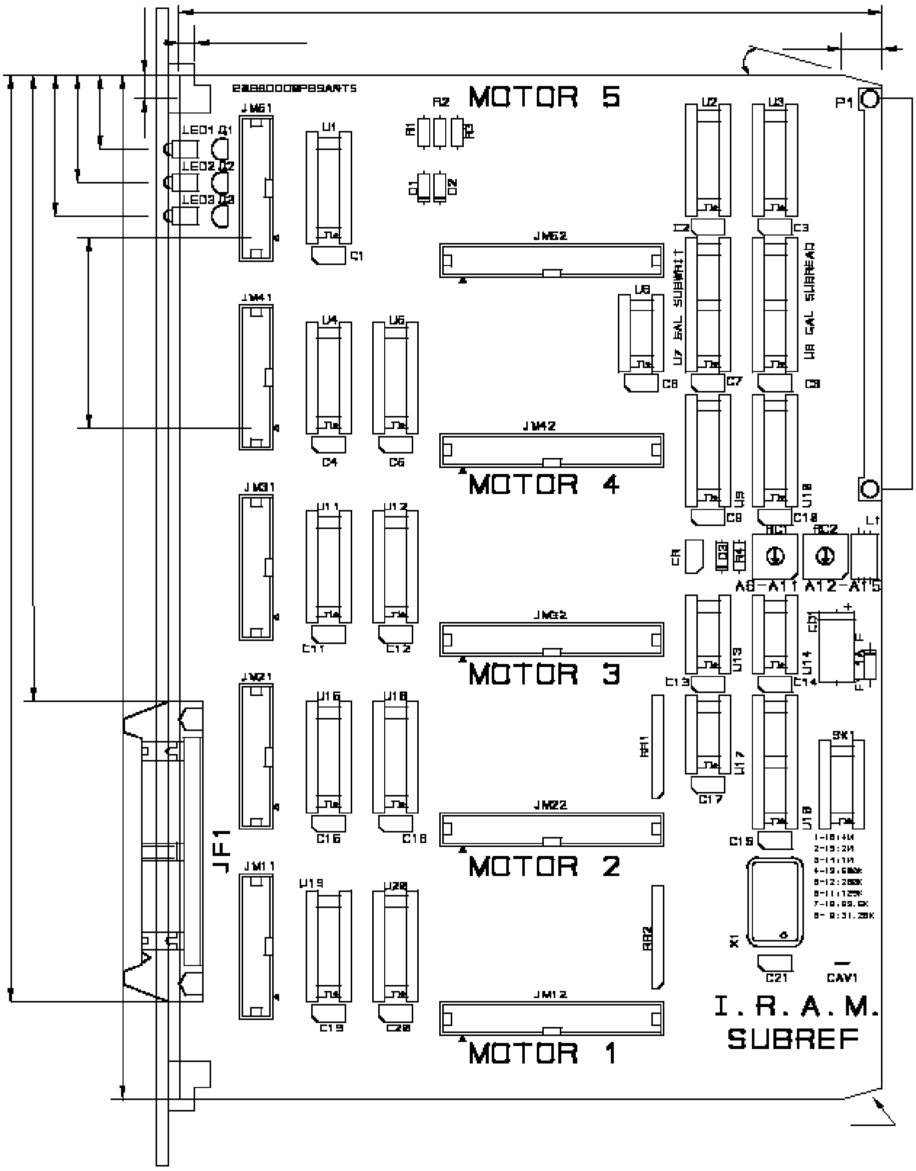
Connections between the Subref board and the SubIsol rack are made through a 40-pos flat cable. All are TTL signals. Each motor needs 5 signals:

Signal name	Signal direction (as viewed from VME)	
INSIN[x]	Input	Motor[x] encoder SIN signal
INCOS[x]	Input	Motor[x] encoder COS signal
/INSWI[x]	Input	Motor[x] switch: 0 Volt when TRUE
OUTUP[x]	Output	Motor[x] “Move Up” command
OUTDN[x]	Output	Motor[x] “Move Down” command

Pinout of the front-panel connector:

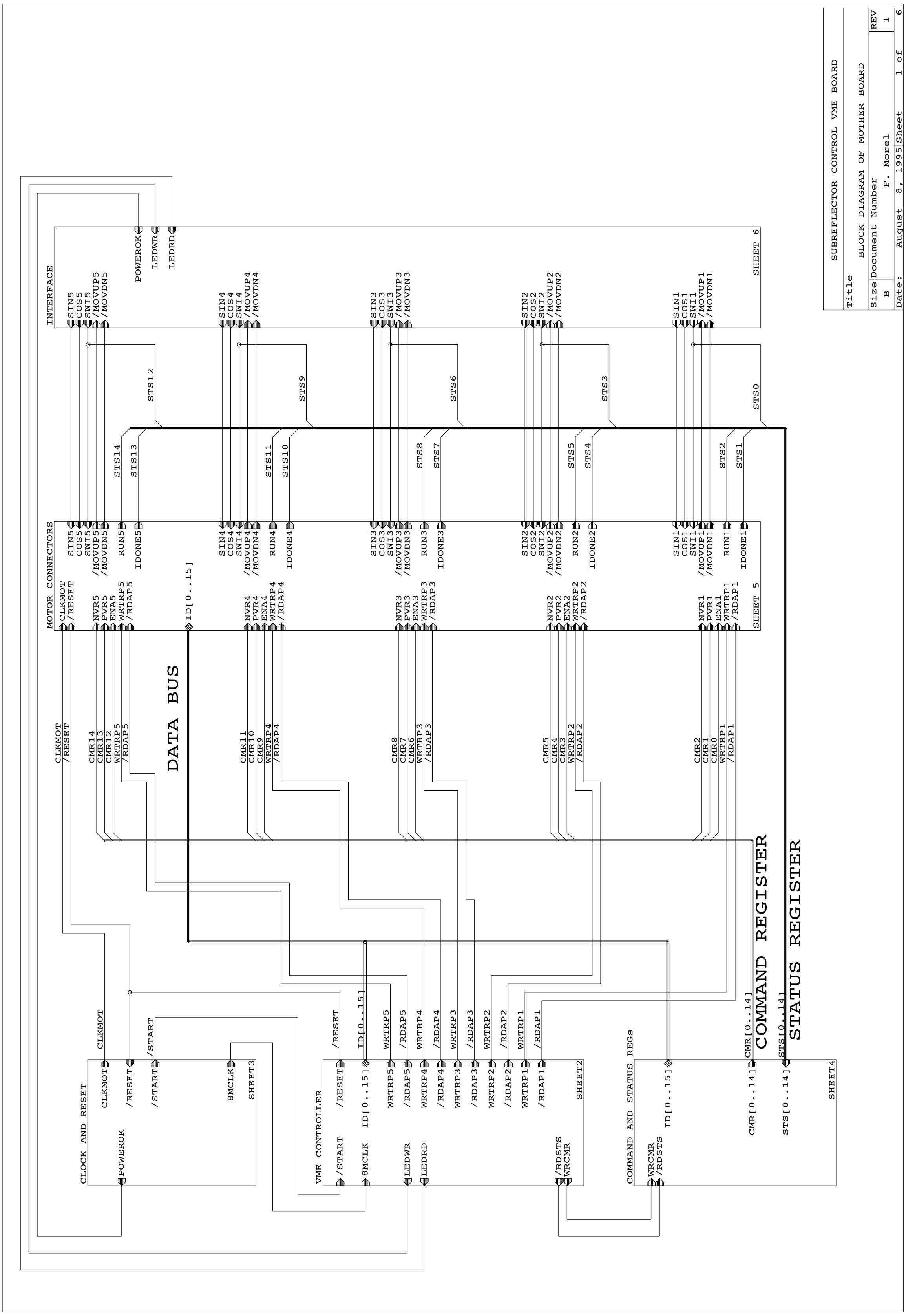
Pin	Signal Name	Comments	Pin	Signal name	Comments
1	+5V Out	Fuse protected	2	+5V Out	Fuse protected
3	+5V Out	Fuse protected	4	+5V Out	Fuse protected
5	+5V Out	Fuse protected	6	+5V Out	Fuse protected
7	+5V Out	Fuse protected	8	+5V Out	Fuse protected
9			10	GND	
11	INSIN1	TTL input	12	INCOS1	TTL input
13	/INSWI1	LOW when switch ON	14	OUTUP1	TTL output
15	OUTDN1	TTL output	16	GND	
17	INSIN2	TTL input	18	INCOS2	TTL input
19	/INSWI3	LOW when switch ON	20	OUTUP2	TTL output
21	OUTDN4	TTL output	22	GND	
23	INSIN3	TTL input	24	INCOS3	TTL input
25	/INSWI3	LOW when switch ON	26	OUTUP3	TTL output
27	OUTDN3	TTL output	28	GND	
29	INSIN4	TTL input	30	INCOS4	TTL input
31	/INSWI4	LOW when switch ON	32	OUTUP4	TTL output
33	OUTDN4	TTL output	34	GND	
35	INSIN5	TTL input	36	INCOS5	TTL input
37	/INSWI5	LOW when switch ON	38	OUTUP5	TTL output
39	OUTDN5	TTL output	40	GND	

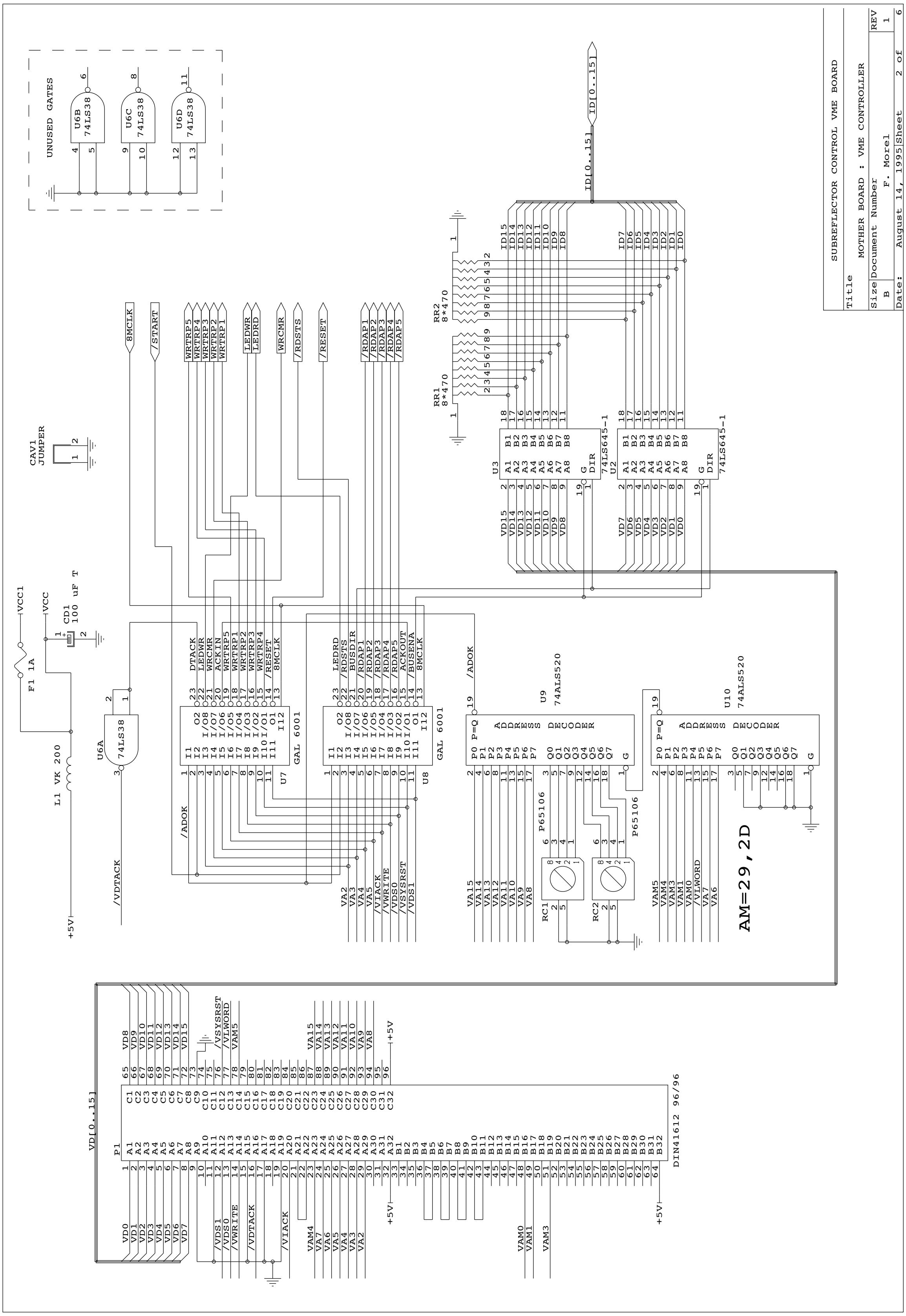
2.3.10 Subref Board layout:

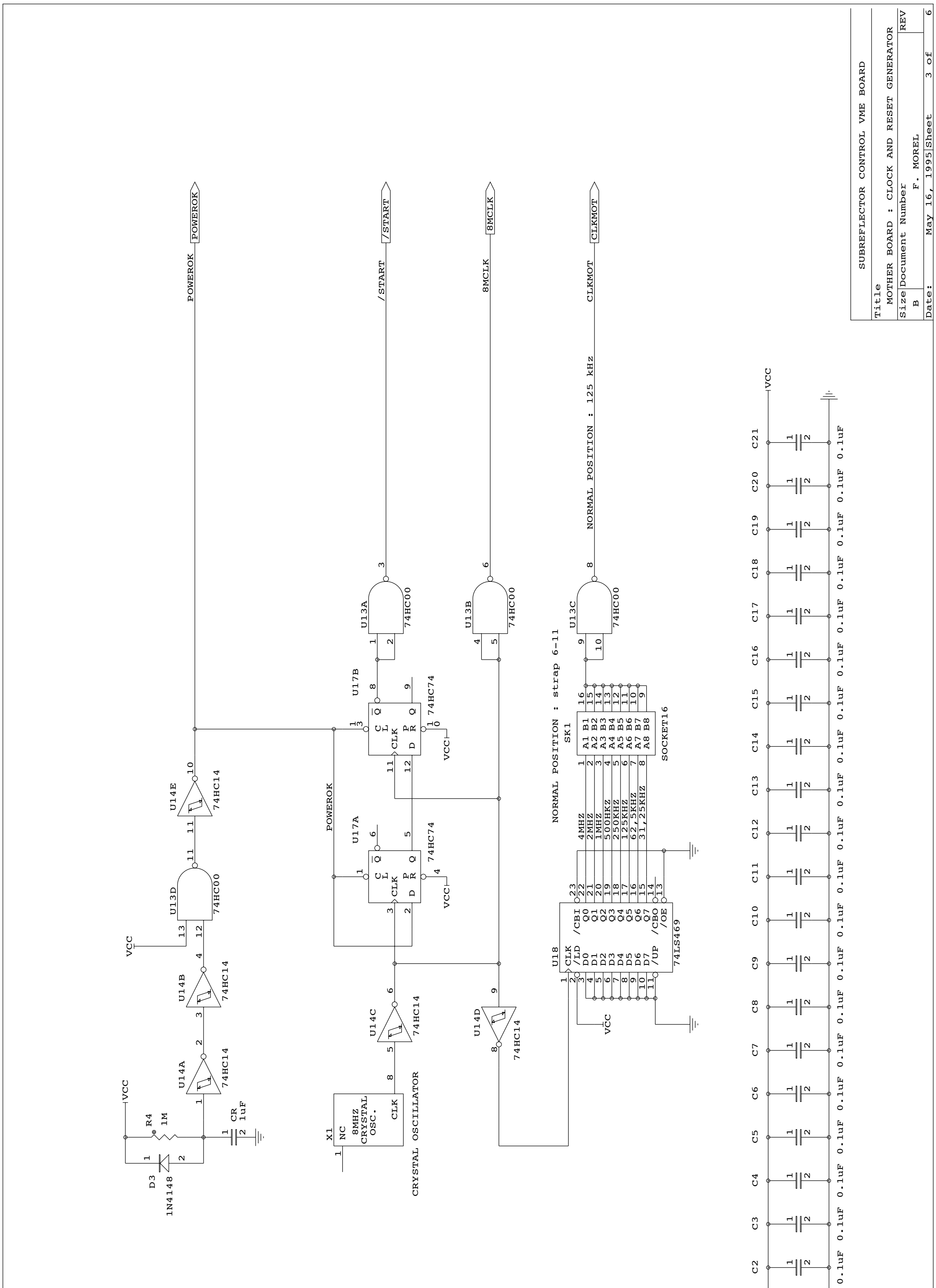


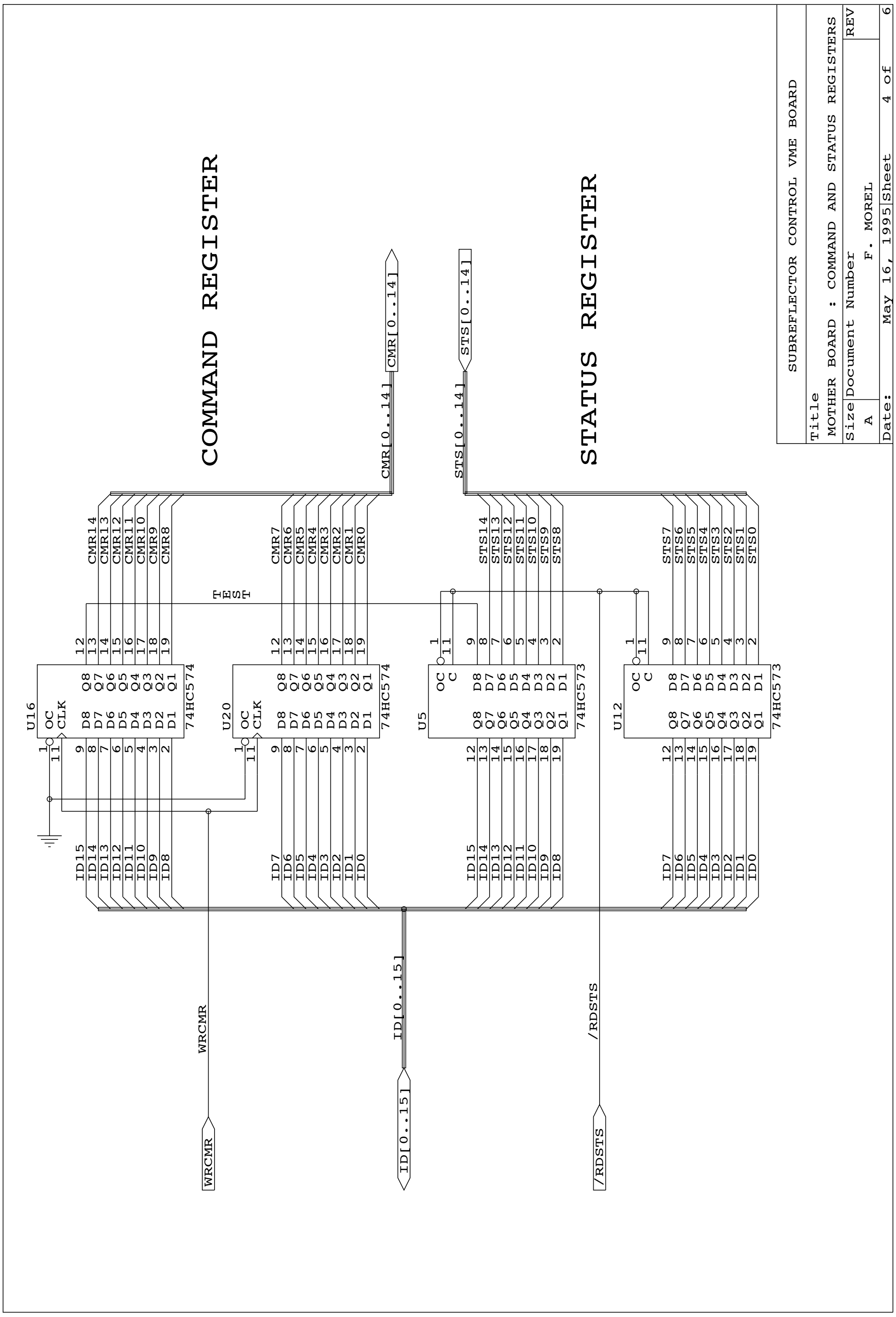
N.B: *as seen with the 5 Submot boards removed*

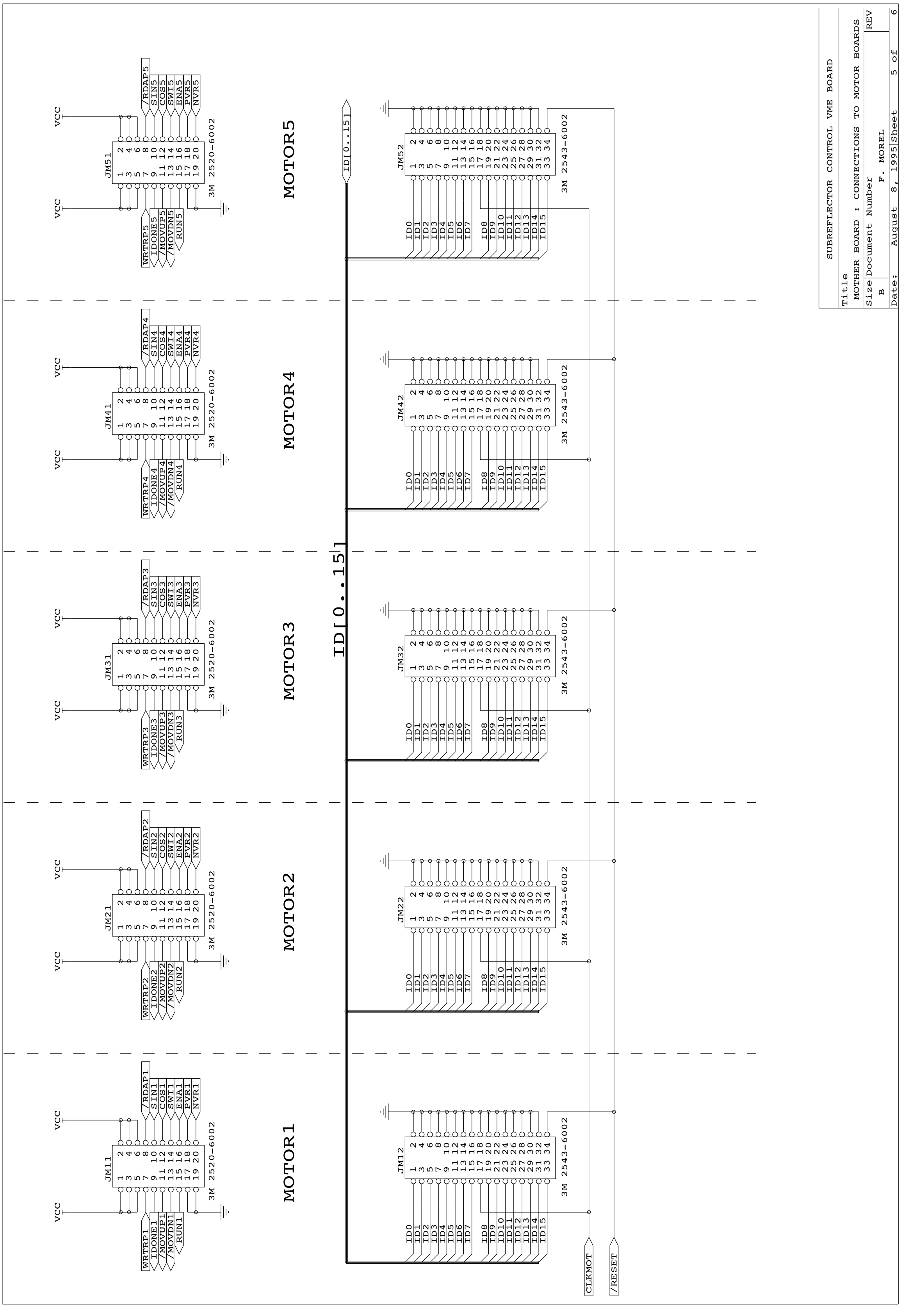
2.3.11 Subref Board schematics:

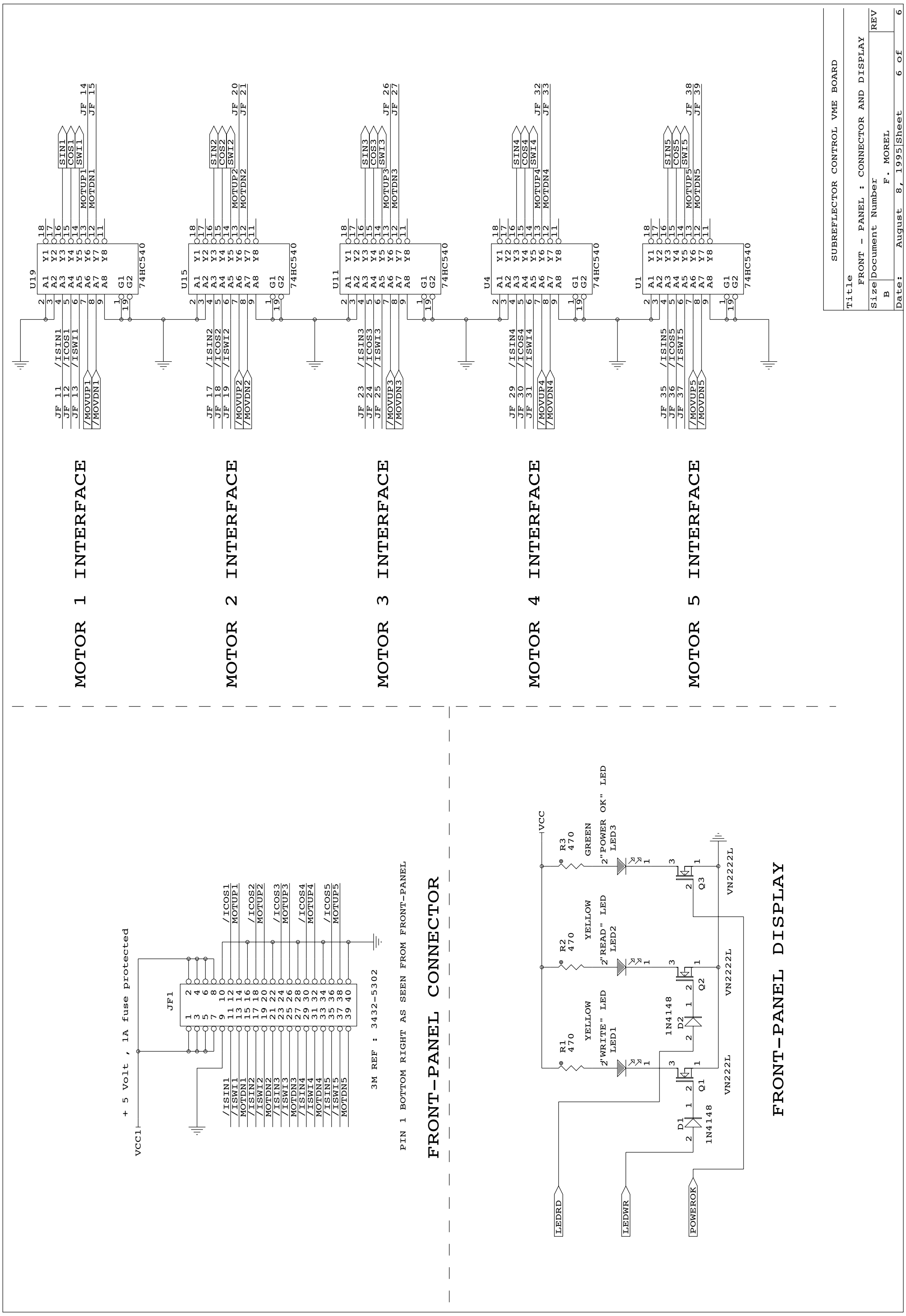



SUBREFLECTOR CONTROL VME BOARD
MOTHER BOARD : VME CONTROLLER
Size B
Document Number F. Morel
REV 1
Date: August 14, 1995
Sheet 2 of 6
AM=29,2D
UNUSED GATES

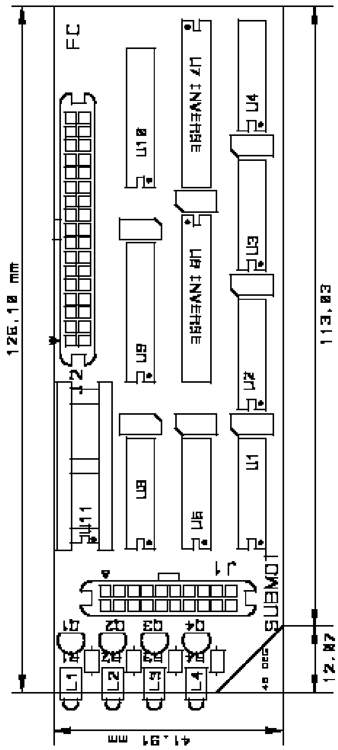

SUBREFLECTOR CONTROL VME BOARD
Title
MOTHER BOARD : CLOCK AND RESET GENERATOR
Size Document Number REV
B F. MOREL
Date: May 16, 1995 Sheet 3 of 6
POWEROK
/START
8MCLK
CLKMOT
NORMAL POSITION : 125 kHz
NORMAL POSITION : strap 6-11
SK1
SOCKET16
4MHZ
2MHZ
1MHZ
500KHZ
250KHZ
125KHZ
62,5KHZ
31,25KHZ
U18
74LS469
U13A
U13B
U13C
U13D
74HC00
U14A
U14B
U14C
U14D
U14E
74HC14
U17A
U17B
74HC74
X1
8MHZ CRYSTAL OSC.
CRYSTAL OSCILLATOR
R4
1M
CR
1uF
D3
1N4148
VCC
0.1uF


COMMAND REGISTER
STATUS REGISTER
CMR[0..14]
STS[0..14]
ID[0..15]
WRCMR
/RDSTS
U16 74HC574
U20 74HC574
U5 74HC573
U12 74HC573
TEST
SUBREFLECTOR CONTROL VME BOARD
Title
MOTHER BOARD : COMMAND AND STATUS REGISTERS
Size Document Number REV
A F. MOREL
Date: May 16, 1995 Sheet 4 of 6

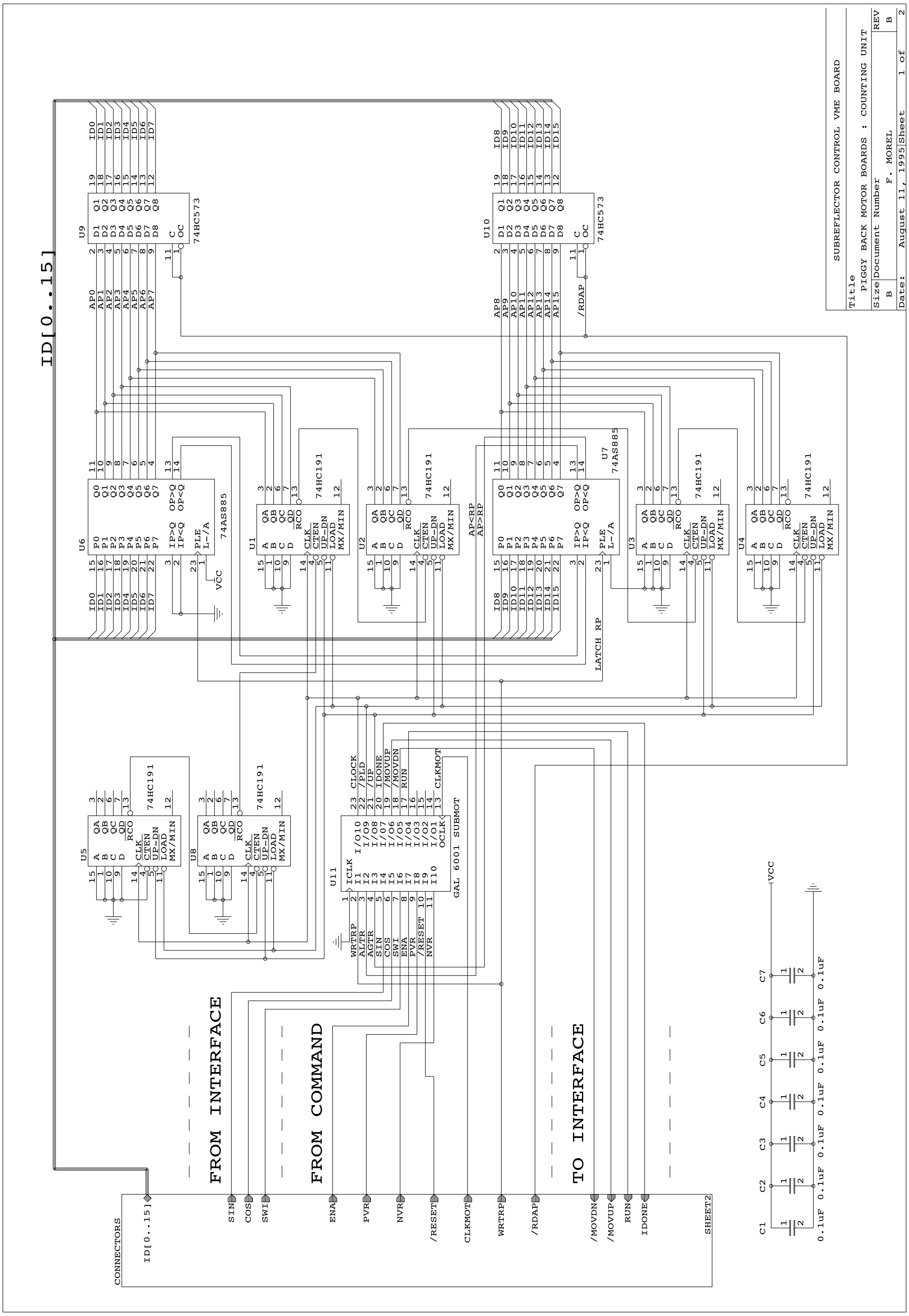

SUBREFLECTOR CONTROL VME BOARD
Title
MOTHER BOARD : CONNECTIONS TO MOTOR BOARDS
Size Document Number REV
B F. MOREL
Date: August 8, 1995 Sheet 5 of 6
MOTOR1
MOTOR2
MOTOR3
MOTOR4
MOTOR5
ID[0..15]
CLKMOT
/RESET
3M 2520-6002
3M 2543-6002

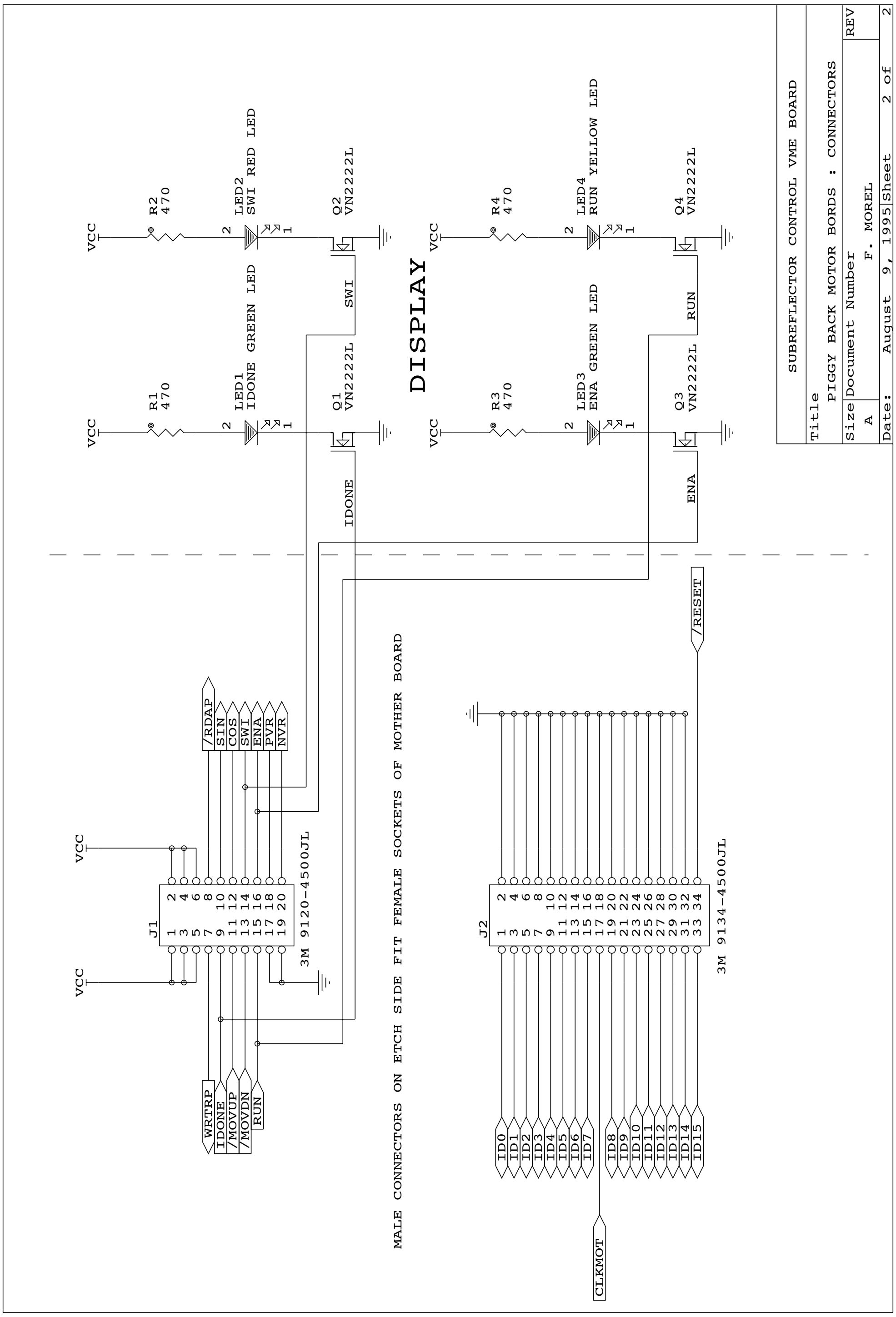

MOTOR 1 INTERFACE
MOTOR 2 INTERFACE
MOTOR 3 INTERFACE
MOTOR 4 INTERFACE
MOTOR 5 INTERFACE
FRONT-PANEL CONNECTOR
PIN 1 BOTTOM RIGHT AS SEEN FROM FRONT-PANEL
3M REF : 3432-5302
+ 5 Volt , 1A fuse protected
FRONT-PANEL DISPLAY
SUBREFLECTOR CONTROL VME BOARD
FRONT - PANEL : CONNECTOR AND DISPLAY
F. MOREL
August 8, 1995
6 of 6

2.3.12 Submot Board layout:



2.3.13 Submot Board schematics:




SUBREFLECTOR CONTROL VME BOARD
Title
PIGGY BACK MOTOR BORDS : CONNECTORS
Size Document Number REV
A F. MOREL 2
Date: August 9, 1995 Sheet 2 of 2
DISPLAY
LED1 IDONE GREEN LED
LED2 SWI RED LED
LED3 ENA GREEN LED
LED4 RUN YELLOW LED
Q1 VN2222L
Q2 VN2222L
Q3 VN2222L
Q4 VN2222L
R1 470
R2 470
R3 470
R4 470
J1 3M 9120-4500JL
J2 3M 9134-4500JL
MALE CONNECTORS ON ETCH SIDE FIT FEMALE SOCKETS OF MOTHER BOARD
/RESET
CLKMOT

3 SOFTWARE:

3.1 Brief description of the CAN protocol used on the Plateau de Bure:

N.B: A detailed description of the CAN protocol used on Plateau de Bure is available as document /netapp1/computer/doc/can/canPdBNG/canPdBNG.pdf.

Each CAN message includes a header. Inside this header, receiving nodes, to decide whether they are concerned with the current message, use 2 fields: The CAN ID (unique identifier on 29 bits), and the DLC (Data Length Count), which declares the number of data bytes of the message. If both these parameters match the values expected by a node, it will accept the message.
Each CAN controller has a unique NODE ID, and uses it to filter the incoming CAN messages.
The CAN2VME Controller accepts 3 kinds of message: Broadcast messages, Control messages and Monitoring messages.

Broadcast messages contain no data. Upon receipt of a Broadcast message, the CAN Controller replies with a message using its own NODE ID as CAN ID.

Control messages must contain at least one byte of data, eventually dummy data if the command is completely defined by the identifier. When receiving a control message, the CAN2VME Controller will reply with an acknowledge message containing NO data, and having the same CAN-ID as the previously received message.

Monitoring messages contain NO data (DLC = 0). When receiving a monitoring message, the CAN2VME Controller replies with a message containing a strictly defined number of data bytes, still using the CAN-ID of the received message.

Normally, only the bus master (on Plateau de Bure, a PC in the antenna) should initiate a transaction, sending a message and expecting a reply. But, to be able to react to VME interrupts in real-time, it was found necessary to send a message which had not been requested: This will be the message "INT_R22_EVENT", sent by the CAN2VME Controller, normally once per second.

3.2 CAN2VME Controller Background task:

The controller normally acts a **slave** device: It receives CAN control/monitoring messages from the CAN master PC, does the requested access to the VME board, and replies with acknowledge/data messages. The incoming CAN messages have higher priority than the background task, triggering the CAN interrupt of the C164. A buffer allows storing up to 16 CAN messages.
When reading a 32-bit data from the VME 22G board, the Controller will automatically access the VME twice in 16-bit mode, and reply with a 4-byte data.
The controller acts as a **master** when the 22G VME board generates a local VME interrupt, synchronized on the "TU01" pulse. In that particular case, the Controller will send the message "INT_R22_EVENT".

4 CAN2VME:

The controller CAN2VME is in charge of the VME boards "22G" and "Subref", and will forward them the commands listed below in chapters 7 and 8.
3 messages are specific to the CAN2VME controller:

4.1 Summary of the Control points:

Name	CAN ID	Data size	Description
SET_CAN2VME_SN	00 08 03 FD	8	Sets the CAN2VME 64-bit Serial Number

SET_CAN2VME_ID	00 08 03 FE	8	Sets the CAN2VME 32-bit NODE_ID
SET_CAN2VME_RESET	00 08 03 FF	1	Resets the CAN2VME controller and the VME Bus

4.2 Control Points in detail:

Name	SET_CAN2VME_SN
CAN ID	00 08 03 FD
Description	Overwrites the Controller Serial Number. Possible only with a 16-bit security key. The Serial Number is stored in a EEPROM and cannot be overwritten when loading a new firmware version.
Data	8 bytes: Bytes [0,1]: 16-bit security key must match 16 MSbits of Serial Number Bytes [2..7]: 48 LSbits of new Serial Number

Name	SET_CAN2VME_ID
CAN_ID	00 08 03 FE
Description	Sets the new NODE_ID of the Controller. Possible only with a 32-bit key. The NODE_ID is stored in a EEPROM and cannot be overwritten when loading a new firmware version.
Data	8 bytes: Bytes [0..3]: 32-bit security key Bytes [4..7]: New NODE_ID

Name	SET_CAN2VME_RESET
CAN ID	00 08 03 FF
Description	Resets the CAN Controller and VME bus. This command executes inside the CAN interrupt routine, and has highest priority. ***Exception***: This control message expects NO acknowledge message.
Data	1 byte (dummy byte to fulfill control message format requirements)

5 CAN22G:

5.1 Summary of Control and Monitor points:

Name	CAN ID	Data Size	Description
GET_R22_CNTR0	00 08 03 00	5	Counter 0 readout
GET_R22_CNTR1	00 08 03 04	5	Counter 1 readout
GET_R22_CNTR2	00 08 03 08	5	Counter 2 readout
GET_R22_PELTIER_T	00 08 03 0C	5	Peltier temp readout
GET_R22_LOAD_T	00 08 03 10	5	Load temp readout
GET_R22_2MHZ	00 08 03 14	5	2 MHz Ref readout
GET_R22_CNTR3	00 08 03 18	5	Counter 3 readout
GET_R22_STATUS	00 08 03 1E	3	Status register readout
SET_R22_CMR	00 08 03 20	1	Command Register write

5.2 Monitor Points in detail:

Name	GET_R22_CNTR0

CAN ID	00 08 03 00
Description	Reads contents of VME board “22G” Counter 0
Data	5 bytes Bytes [0..3] = 32-bit unsigned value of counter 0. Byte [4]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_R22_CNTR1
CAN ID	00 08 03 04
Description	Reads contents of VME board “22G” Counter 1
Data	5 bytes Bytes [0..3] = 32-bit unsigned value of counter 1 Byte [4]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_R22_CNTR2
CAN ID	00 08 03 08
Description	Reads contents of VME board “22G” Counter 2
Data	5 bytes Bytes [0..3] = 32-bit unsigned value of counter 2 Byte [4]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_R22_PELTIER_T
CAN ID	00 08 03 0C
Description	Reads Peltier cooler temperature of the 22G Receiver
Data	5 bytes Bytes [0..3] = 32-bit unsigned value of Peltier cooler temperature Byte [4]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_R22_LOAD_T
CAN ID	00 08 03 10
Description	Reads Load temperature of the 22G Receiver
Data	5 bytes Bytes [0..3] = 32-bit unsigned value of Load temperature Byte [4]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_R22_2MHZ
CAN ID	00 08 03 14
Description	Reads the 2 MHz Reference of the 22G Receiver
Data	5 bytes Bytes [0..3] = 32-bit unsigned value of the 2 MHz frequency Byte [4]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_R22_CNTR3
CAN ID	00 08 03 18
Description	Reads contents of VME board “22G” Counter 3
Data	5 bytes Bytes [0..3] = 32-bit unsigned value of counter 3 Byte [4]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_R22_STATUS
CAN ID	00 08 03 1E
Description	Reads the 22G Receiver Status
Data	3 bytes: Bytes [0,1] = 16-bit unsigned value of Status. Byte [0]: Bit [7]: ERR, set if any error occurs. Bit [6..3]: unused Bit [2]: CAN error Bit [1: VME bus time-out Bit [0]: VME bus stuck Byte [1]: Bit [7,6]: unused Bit [5]: 22G receiver ALARM Bit [4]: UNL, the 22G VME board is unlocked, and no longer synchronized with the “TU01” pulse. Bit [3]: IT_ENA, the VME interrupt is enabled Bit [2]: NOISE_ON, the noise diode has been requested to turn ON (This bit is NOT read from the receiver) Bit[1]: LOAD_ON, the reference load is in front of the receiver Bit[0]: unused Byte [2]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

5.3 Control Points in detail:

Name	SET_R22_CMR
CAN ID	00 08 03 20
Description	Writes the 22G Receiver Command register
Data	1 byte: Byte [0]: Bit [7..4]: unused Bit [3]: CMD_IT_ENA, enables the VME interrupt Bit [2]: CMD_NOISE_ON, turns the noise diode ON Bit [1]: CME_LOAD_ON, moves the load in front of the receiver. Bit [0]: CMD_PWR, unused but functional

5.4 Time Event message:

Name	INT_R22_EVENT
CAN ID	00 08 03 FC
Description	Sent by the CAN2VME Controller, it reports that the VME board generated an

	interrupt. This interrupt occurs normally once per second, triggered by the "TU01" pulse received from the GPS time-base distribution.
Data	1 byte data Data = 0, OK Data = 1, Error: The VME 22G board has lost synchro with the "TU01". Data = 2, Error: The VME 22G board did not acknowledge the interrupt.

6 CANSubref

6.1 Summary of Control and Monitor points:

Name	CAN ID	Data Size	Description
GET_SUBREF_STATUS	00 08 02 00	3	Status register
GET_SUBREF_MOTOR1	0 008 02 04	3	Motor 1 actual position readout
GET_SUBREF_MOTOR2	00 08 02 08	3	Motor 2 actual position readout
GET_SUBREF_MOTOR3	00 08 02 0C	3	Motor 3 actual position readout
GET_SUBREF_MOTOR4	00 08 02 10	3	Motor 4 actual position readout
GET_SUBREF_MOTOR5	00 08 02 14	3	Motor 5 actual position readout
SET_SUBREF_COMMAND	00 08 02 20	2	Command register
SET_SUBREF_MOTOR1	00 08 02 24	2	Motor 1 reference position
SET_SUBREF_MOTOR2	00 08 02 28	2	Motor 2 reference position
SET_SUBREF_MOTOR3	00 08 02 2C	2	Motor 3 reference position
SET_SUBREF_MOTOR4	00 08 02 30	2	Motor 4 reference position
SET_SUBREF_MOTOR5	00 08 02 34	2	Motor 5 reference position

6.2 Monitor Points in detail:

Name	GET_SUBREF_STATUS
CAN ID	00 08 02 00
Description	Get status register
Data	3 bytes Byte [0]: Bit [7]: Test Bit [6]: RUN5, 1: motor 5 is running. Bit [5]: IDONE5, 1: motor 5 init done. Bit [4]: SW5, init switch status. 1: end of motor 5 stroke, 0: most of its stroke. Bit [3]: RUN4, 1: motor 4 is running. Bit [2]: IDONE4, 1: motor 4 init done. Bit [1]: SW4, init switch status. 1: end of motor 5 stroke, 0: most of its stroke. Bit [0]: RUN3, 1: motor 3 is running. Byte [1]: Bit [7]: IDONE3, 1: motor 3 init done. Bit [6]: SW3, init switch status. 1: end of motor 5 stroke, 0: most of its stroke. Bit [5]: RUN2, 1:motor 2 is running. Bit [4]: IDONE2, 1: motor 2 init done. Bit [3]: SW2, init switch status. 1: end of motor 5 stroke, 0: most of its stroke. Bit [2]: RUN1, 1:motor 1 is running. Bit [1]: IDONE1, 1: motor 1 init done. Bit [0]: SW1, init switch status. 1: end of motor 5 stroke, 0: most of its stroke.

	IDONE* gets equal to 1 when ENA*=1 (see control register) and when SW* switches from 0 to 1. Byte [2]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_SUBREF_MOTOR1
CAN ID	00 08 02 04
Description	Get motor 1 actual position
Data	3 bytes Bytes [0,1]: Reading, 2's complement signed value. Equal to 0 at start time . Byte [2]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_SUBREF_MOTOR2
CAN ID	00 08 02 08
Description	Get motor 2 actual position
Data	3 bytes Bytes [0,1]: Reading, 2's complement signed value. Equal to 0 at start time. Byte [2]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_SUBREF_MOTOR3
CAN ID	00 08 02 0C
Description	Get motor 3 actual position
Data	3 bytes Bytes [0,1]: Reading, 2's complement signed value. Equal to 0 at start time. Byte [2]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_SUBREF_MOTOR4
CAN ID	00 08 02 10
Description	Get motor 4 actual position
Data	3 bytes Bytes [0,1]: Reading, 2's complement signed value. Equal to 0 at start time. Byte [2]: Transaction report Bit [2]: CAN error Bit [1]: VME time-out Bit [0]: VME Bus stuck

Name	GET_SUBREF_MOTOR5
CAN ID	00 08 02 14
Description	Get motor 5 actual position
Data	3 bytes Bytes [0,1]: Reading, 2's complement signed value. Equal to 0 at start time. Byte [2]: Transaction report Bit [2]: CAN error

	Bit [1]: VME time-out Bit [0]: VME Bus stuck

6.3 Control Points in detail:

Name	SET_SUBREF_COMMAND
CAN ID	00 08 02 20
Description	Set command register
Data	2 bytes Byte [0]: Bit [7]: Test Bit [6]: NVR5, motor 5 negative velocity request. Bit [5]: PVR5, motor 5 positive velocity request. NVR5=PVR5=1: Stop motor 5. NVR5=PVR5=0: Request to move to motor 5 reference position. Bit [4]: ENA5, enable motor 5 init and then enable position request. Bit [3]: NVR4, motor 4 negative velocity request. Bit [2]: PVR4, motor 4 positive velocity request. NVR4=PVR4=1: Stop motor 4. NVR4=PVR4=0: Request to move to motor 4 reference position. Bit [1]: ENA4, enable motor 4 init and then enable position request. Bit [0]: NVR3, motor 3 negative velocity request. Byte [1] Bit [7]: PVR3, motor 3 positive velocity request. NVR3=PVR3=0: Stop motor 3. NVR3=PVR3=1: Request to move to motor 3 reference position. Bit [6]: ENA3, enable motor 3 init and then enable position request. Bit [5]: NVR2, motor 2 negative velocity request. Bit [4]: PVR2, motor 2 positive velocity request. NVR2=PVR2=1: Stop motor 2. NVR2=PVR2=0: Request to move to motor 2 reference position. Bit [3]: ENA2, enable motor 2 init and then enable position request. Bit [2]: NVR1, motor 1 negative velocity request. Bit [1]: PVR1, motor 1 positive velocity request. NVR1=PVR1=1: Stop motor 1. NVR1=PVR1=0: Request to move to motor 1 reference position. Bit [0]: ENA1, enable motor 1 init and then enable position request.

Name	SET_SUBREF_MOTOR1
CAN ID	00 08 02 24
Description	Set motor1 reference position
Data	2 bytes Bytes [0,1]: Reading, 2's complement signed value.

Name	SET_SUBREF_MOTOR2
CAN ID	00 08 02 28
Description	Set motor2 reference position
Data	2 bytes Bytes [0,1]: Reading, 2's complement signed value.

Name	SET_SUBREF_MOTOR3
CAN ID	00 08 02 2C
Description	Set motor3 reference position
Data	2 bytes Bytes [0,1]: Reading, 2's complement signed value.

Name	SET_SUBREF_MOTOR4
CAN ID	00 08 02 30
Description	Set motor4 reference position
Data	2 bytes Bytes [0,1]: Reading, 2's complement signed value.

Name	SET_SUBREF_MOTOR5
CAN ID	00 08 02 34
Description	Set motor5 reference position
Data	2 bytes Bytes [0,1]: Reading, 2's complement signed value.