PCI-Bus to CAMAC Interface
Type: **PCI-CC32**
Version 1.0

ARW - Elektronik
Schlehenweg 1
D-69168 Wiesloch
Germany
Tel. 06222/50816

Jan. 2000: Erstellung des Dokuments
Feb. 2000: 1. Überarbeitung
Nov. 2001: 2. CC32 + ReadDoubleWord

Legende

I	=	**Lötbrücke gesteckt**
:	=	**Lötbrücke offen**

Absolute Adressen werden hexadezimal entweder mit einem führenden „$“- Zeichen oder mit einer führendem „0x“ wie in C-Notation üblich dargestellt.

Beispiele: $00AA entspricht 0x00AA entspricht dezimal 170.

„don’t care“ Bedingungen werden mit einem „X“ verdeutlicht. Dann ist der Inhalt unbedeutend.

Inhaltsverzeichnis

1 VORWORT

Mit dem von der Firma ARW-Elektronik entwickelten PCI-CAMAC Interface **CC32** kann sehr effizient und schnell vom einem PC (welcher mit PCI Steckplätzen ausgestattet ist) direkt auf den CAMAC-BUS zugegriffen werden. Es sind D16 und D32 breite Datentransfers vom PCI-BUS zum CAMAC-CONTROLLER möglich.

Das Interface besteht aus einer PCI-Adapter-Karte (nachfolgend PCIADA genannt), und dem CAMAC-CONTROLLER Modul (nachfolgend CC32 genannt).

Für den Kompakt-PCI-Bus ist eine Karte (CPCIADA) mit den gleichen Eigenschaften von PCIADA verfügbar.

Auch eine VME-Slave Karte (VMEADA) ist verfügbar.

Die Datenübertragung erfolgt mittels differentiellen Low-Level-Leitungstreiber (LVDS) und einem 50pol. geschirmten Kabel von bis zu 3 m Länge. Das Kabel ist ein handelsübliches SCSI2-Kabel mit 50 pol. Feinpitch Stecker auf beiden Seiten. Optional sind Verlängerungskabel von jeweils 3 m lieferbar. Für eine zuverlässige Übertragung sollte die Gesamtlänge 10 m Distanz zwischen den beiden Adaptern nicht überschreiten. Durch die differentielle Übertragungstechnik und durch das mit „Hardware-Handshake“ verriegelte parallele Übertragungsverfahren sind hohe Übertragungsraten bei gleichzeitig stark verbessertem Störverhalten erreicht worden.

Obwohl die Beschreibung sich stark an der PC-Technik und den MS-Betriebssystemen anlehnt, ist das Interface für alle Rechner mit 33 MHz PCI-BUS verwendbar. Auch ist als zugehöriges Host-Betriebssystem jedes andere auf solchen Rechnern lauffähige Betriebssystem denkbar. (Beispiele: PowerPC, Sparc, Alpha und LINUX, UNIX, MacOS, Solaris etc.)

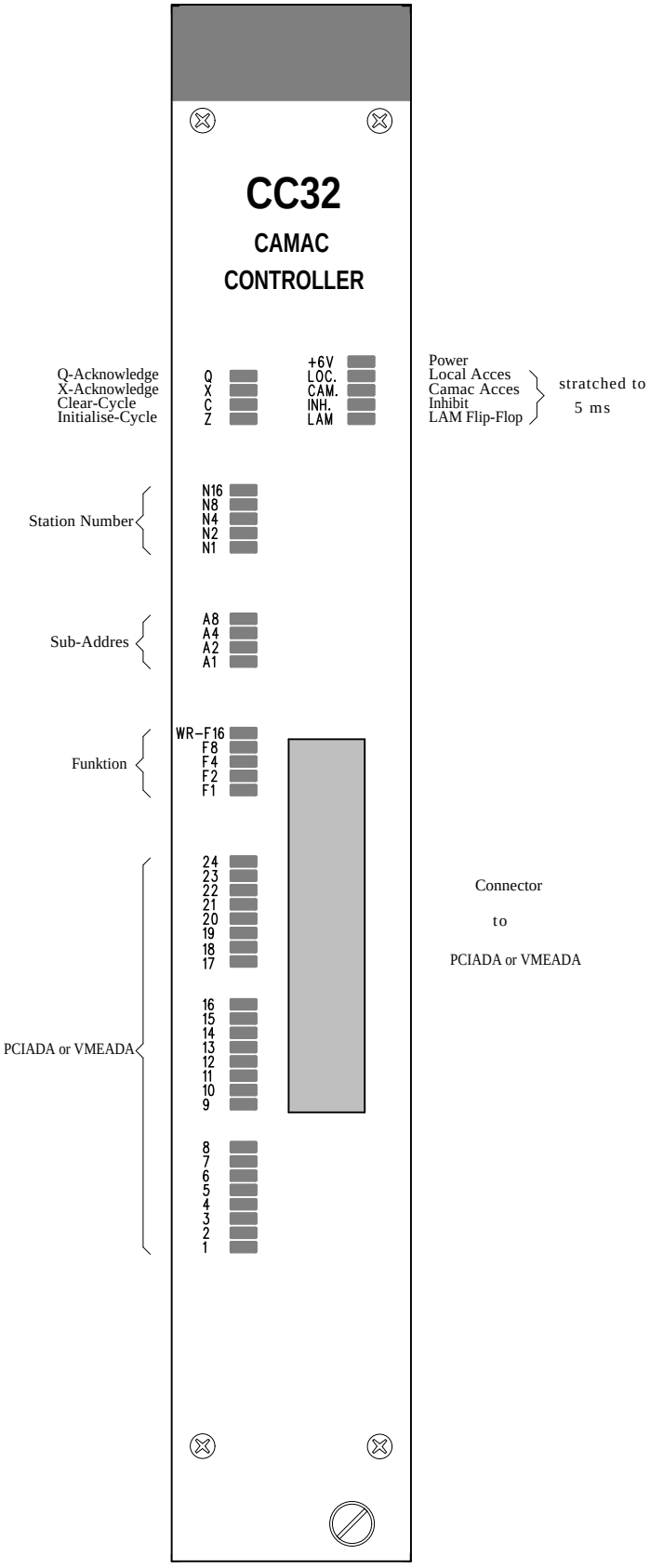
1.1 Eigenschaften PCIADA

- PCI-Interface mit Standard PCI Baustein **PLX9050** von *PLX-Technology*
  siehe Datenblatt auf CD unter....\PCIADA\CHIPDOCU\PLX9050.PDF
- Es werden (8), 16 u. 32 Bit PCI-BUS Slave Zugriffe unterstützt.
- Direkte Datenabbildung in den Zieladressbereich mit automatischer „low-big-endian“ Konversion.
- Das Interface belegt **32K** Speicher-Adressbereich unterhalb von 1MB und ist damit auch unter MS-DOS adressierbar.
- Es wird je eine maskierbare Interruptquelle von CC32 und eine lokale „Timeout“-Interruptquelle unterstützt.
- Die Verwendung differentieller Signaltreiber und Empfänger bürgt für störsichere u. schnelle Datenübertragung.
- **Neu!** AutoRead PCIADA kann selbständig das nächste Datum aus dem CC32 lesen

1.2 Eigenschaften CC32

- Transparenter D16 u. D24 (D32) CAMAC-DATAWAY-Zugriff (ohne Hilfsregister)
- 32K Adressabbildung nach **NAF**.
- 24 bit programmierbares LAM-Maskenregister
- LAM-Interrupt zu PCIADA
- FASTCAMAC Level 1
- LED-Anzeige für:
  +5V, CAMAC-Zugriff, lokaler CC32-Zugriff, INHIBIT u. LAM. (min 5 ms)
  Q, X, C, Z, N1..16, A1..8, F1..16 u. DATA1..24.
- CAMAC-Zyklusszeit von Busy bis zu S1 für jede Station modifizierbar auf
  300 ns u. 200ns, S1 u. S2 = 100ns
- Broadcast CAMAC-WRITE + Broadcast-Maskenregister
- Keine Beeinflussung des PCs beim Ein- oder Ausschalten der CAMAC-Seite (Bedingung: Interrupt disabled). Beim Ein- oder Ausschalten des PCs wird im CC32 ein RESET aktiviert.
- **Neu!** CC32 kann bei einem Camac-Read gleich zwei 16-bit Datenworte lesen.

1.3 Frontansicht CC32



1.4 Installation PCIADA + CC32

1.4.1 Allgemeine Hinweise

Hinweis: Sorgen sie vor dem Auspacken für eine elektrostatisch entladene Arbeitsumgebung. Fassen sie nie direkt auf die elektronischen Bauteile. Elektrostatische Entladung kann zu Schäden an den elektronischen Bauteilen führen.

Hinweis: Versichern sie sich, daß sich der Quellrechner und der CAMAC-Einschubrahmen auf gleichem Spannungspotential befinden. Unerwartete Ausgleichsströme können zu Schäden an den elektronischen Bauteilen führen.

Hinweis: Die Karten dürfen nie bei eingeschalteter Spannungsversorgung eingesteckt oder ausgezogen werden. Unerwartete Seiteneffekte bei Einstecken unter Spannung können elektronische Bauteile zerstören.

1.4.2 Schrittweise Installation

1. Vor der Installation sind die Steckbrücken des CC32 zu prüfen bzw. entsprechend der gewünschten Funktion zu ändern. (siehe **1.5. Modulnummer**). Auf der PCIADA-Karte gibt es keine Benutzer-konfigurierbare Steckbrücken.
2. Stecken sie die PCIADA-Karte in einen verfügbaren PCI-Steckplatz des PCs und schrauben sie die Karte fest.
3. Stecken sie CC32 in den CAMAC-Einschubrahmen (ganz rechte Station) und schrauben sie ihn mittels der Sicherungsschraube an der Frontplatte fest.
4. Schließen sie das 50 pol. Verbindungskabel zwischen PCIADA und VCC32 an.
5. Schalten sie sowohl den PC als auch den CAMAC-Einschubrahmen ein. Beim erstmaligen Starten von Windows95 erscheint die Meldung, daß eine neue Hardware erkannt wurde. Aktivieren sie „keinen Treiber installieren (keine erneute Aufforderung zur Installation)“ und bestätigen dies mit „OK“.
6. Installieren sie entsprechend dem Handbuch PCI-CC32 WIN95 Treiber die Software.
7. Für die Pascal - Software müssen sie in CONGIG.SYS den EMM386.EXE (entsprechend der Beschreibung unter A:\DOS\PASCAL\Readme.txt) einbinden.
8. Testen sie die Installation mit der mitgelieferten Prüf-Software.

1.5 Modulnummer

Die Position der Steckbrücken ist dem Bestueckungsplan CC32-CS zu entnehmen. Siehe 3.17

Mit dieser einstellbaren Nummer kann die Software eines PCs mehrere gleichzeitig angeschaltete CC32 Camac Controller unterscheiden.

Hinweis: Bei gleicher Einstellung mehrerer CC32 Controller kann nicht zwischen den Anschaltungen (CRATEs) unterschieden werden. Dies kann nicht vorhersagbare Effekte zur Folge haben.

Die Einstellung kann über **CC32-Status** gelesen werden.

J304	J303	J302	J301	Funktion
X	X	X	X	Kennung bei Multi PCI-CC32 installation
I	I	I	:	(werkseitige Einstellung, Nummer = 1)

1.6 Zugriffszeiten

Typische Zugriffszeiten mit einem Pentium 500MHz (Windows98) einschließlich Progamm-Code lesen und 32-bit Datum in ein Array schreiben:

Zugriff	WR-time / us	RD-time / us
PCI to CC32 D16-intern	0,6	0,9
PCI to CC32 D32-intern	0,65	1,3
PCI to CC32 D16-Camac	1,4 (1,2)	1,6 (1,4)
PCI to CC32 D32-Camac	1,4 (1,2)	1,8 (1,6)
PCI to CC32 D32-Camac AutoRead	-	1,1 (0,9)
PCI to CC32 D32-Camac FCL1 + AutoRead	-	0,7
PCI to CC32 2*D16-Camac FCL1 + AutoRead + RDW	-	0,9

Werte in (..) mit Cycle-Tunebits = 11

2 PCIADAPTER KARTE

2.1 32K Adressbereich

Hinweis: PCIADA ist für den Betrieb in Verbindung mit CC32 auf einen 32k Adressbereich konfiguriert im Gegensatz zu 8K für den Betrieb mit dem VMEMasterModul (VMEMM). Es werden anstatt der Adressbits A0 u. A1 jetzt die Adressbits A13 u. A14 übertragen. In Verbindung mit CC32 ist das EEPROM auf PCIADA mit einem **C** beschriftet.

Im Bedarfsfall kann das EEPROM vom Benutzer im Rechner selbst umprogrammiert werden. Es besteht auch die Möglichkeit PCIADA für die wechselweise Verwendung mit CC32 und VMEMM zu konfigurieren.

Die Anweisung hierzu finden sie auf der beiliegenden CD in WORKAROUND.TXT

2.2 Übersicht/Funktion

Hinweis: Im weiteren wird vorausgesetzt, daß der verwendete PC ein PCI-BIOS mit seinen Autokonfigurationsfunktionen besitzt. Manche ältere Rechner weisen diese Eigenschaft nicht auf.

Um eine sichere Anpassung an den PCI-Bus zu gewährleisten, wurde der PCI-Bus Target-Interface-Chip **PLX9050** der Fa. *PLX-Technology* ausgewählt. Die PCIADApterkarte konfiguriert sich entsprechend dem Inhalt des **PCI CONFIGURATION REGISTER** (**PCR**) beim Booten des Rechners selbst. Die Inhalte des **PCR**egisters sind in einem EEPROM abgelegt und werden beim Einschalten des PCs automatisch in den **PCI9050** übernommen. Es werden 3 unterschiedliche Adressbereiche von PCIADA angefordert. Diese sind wie folgt:

1. 54 Byte **LOCAL CONFIGURATION REGISTER** (**LCR**) im I/O-Bereich. Im **LCR** sind die Basisadressen, sowie diverse Control-Status-Register verfügbar. (siehe Datenblatt **PCI9050**)

   Hinweis: Das LCR im I/O-Bereich ist nicht auf allen Rechnerplattformen mit PCI-Bus verfügbar. Es sollte dann die inhaltsgleichen LCRegister im Memory-Bereich verwendet werden.
2. **32Kbyte Memory-Bereich** für Zugriff auf den lokalen CC32-Adressraum und den direkten Durchgriff auf den CAMAC-DATAWAY.

2.3 AutoRead Function

Mit Hilfe der AutoRead Funktion kann man die Lesezeit der Daten aus dem CC32 reduzieren. Durch beschreiben des **LC-Registers** User0-bit wird die Funktion ein- bzw. ausgeschaltet.

Der AutoRead wird eingeschaltet durch das rücksetzen des **LC-Registers** User0-bit = 0. Der erste Read wird normal ausgeführt mit Übergabe der NAF-Werte zum CC32. Nach dem Ende des PCI-Read liest PCIADA ohne NAF-Übergabe selbständig aus dem CC32 das nächste Datum und hält es für den nächsten PCI-Read bereit. Hierdurch wird die Lesezeit um ca. 500 -700 ns verringert

Der AutoRead wird beendet, wenn das User0-bit = 1 gesetzt wird. Ein bereits gelesenes Datum wird verworfen. Wird das User-bit sofort wieder mit „0“ beschrieben, wird ein neuer AutoRead (mit neuem NAF-Wert gestartet.

Ein Write beendet ebenfalls den AutoRead und verwirft ein bereits gelesenes Datum. PCIADA geht jedoch nach einem Read wieder in den AutoRead Modus.

2.4 Wichtige PCRegister

Zum Identifizieren der Hardware erfragt das PCI-BIOS bzw. die Software-Treiber die Konfiguration der PCIADA Karte. Die ID sind wie folgt festgelegt:

ID	PCIADA Defined value for CC32	**PCIADA Defined value for VMEMM**
Vendor ID	$10B5	$10B5
Device ID	$2258	$9050
Subsystem Vendor ID	$9050	$9050
Subsystem Device ID	$2258	$1167

2.5 Weitere wichtige LCRegister

Andere als die hier erklärten Register müssen in dem Baustein nicht programmiert werden. Die Basisadressen der LCRegister und des CC32 Adressbereichs werden gemäß der PCI-Spezifikation beim Booten des PCs dynamisch zugeteilt. Sie müssen aus dem PLX-Baustein gelesen oder über das PCI-Bios erfragt werden (siehe 4.x Software). Sonstige Funktionen sind dem Datenblatt des PCI9050 Bausteins zu entnehmen.

Siehe CD ..PCIADA\Chipdocu\9050-1ds.pdf

2.5.1 Interrupt Control/Status Register

Das INTERRUPT1 Statusbit wird von CC32 (LAM-FF) erzeugten Interrupts auf 1 gesetzt. Hierzu müssen die zugehörigen Interrupt-Anfragen zugelassen sein. Folgende Interrupt-Quellen werden hier zusammengefaßt:

1. Interrupt durch gesetztes LAM-FF.

Das INTERRUPT2 Statusbit wird durch lokale Ereignisse der PCIADA aktiv. Wie fogt:

1. Zugriff bei nicht angeschlossenem Datenkabel.
2. Zugriff bei nicht eingeschaltetem CAMAC-Einschubrahmen.
3. Wenn das PCI-CC32 Interface nicht freigeschaltet ist. (Das Bit USER I/O2 Output muß vor einem Zugriff auf CC32 gesetzt sein!).

INTCSR; LCR-Baseadr + $4c (by word-access)

Byte		RD	WR	after Init
0	Local Interrupt 1 enable / 1 = enable / 0 = disable / Quelle = CC32	yes	yes	1
1	Local Interrupt 1 polarity / 1 = activ high / 0 = activ low	yes	yes	0
2	Local Interrupt 1 status / 1 = active. 0 = not active	yes	no	0
3	Local Interrupt 2 enable / 1 = enable / 0 = disable / Quelle = PCIADA	yes	yes	1
4	Local Interrupt 2 polarity / 1 = activ high / 0 = activ low	yes	yes	0
5	Local Interrupt 2 status / 1 = active. 0 = not active	yes	no	0
6	PCI Interrupt enable / 1 = enable / 0 = disable (Quelle = global)	yes	yes	0
7	Software Interrupt / 1 = generate Interrupt	yes	yes	0
15..8	not used	yes	no	00000000 b

2.5.2 User I/O Register / Reset Interrupt

USER I/O2:

1. Um zu vermeiden, daß beim Booten des PCs bzw. beim Starten des PC-Betriebssystems in den CC32-Bereich gegriffen wird, wird der USER I/O2 Ausgang zum Sperren des CC32-Bereichs benutzt.
2. Ein gelöschtes Bit „USER I/O2“ setzt auch den „Interrupt 2“ zurück.

USER I/O3:

Ein gesetztes Bit „USER I/O3“ zeigt an, daß der CAMAC-Einschubrahmen eingeschaltet und das Verbindungskabel gesteckt ist.

CNTRL; LCR-Baseadr + $50 (by word-access)

Byte		RD	WR	nach Init
5..0	000100b or 000110b autoread = off / 000010b autoread = on	yes	yes	000110 b
6	USER I/O2 Type, must always be 0	yes	yes	0
7	USER I/O2 Direction, must always be 1 = output	yes	yes	1
8	USER I/O2 output, 0 = disable access to CC32, 1 = enable access	yes	yes	0
9	USER I/O3 Type, must always be 0	yes	yes	0
10	USER I/O3 Direction, must always be 0 = input	yes	yes	0
11	USER I/O3 Input, 0 = CC32 failed , 1 = CC32 OK	yes	no	0
15..12	do not modify	yes	yes	0100 b

Write 0x4086 for disable access to CC32
Write 0x4186 for enable access to CC32 without autoread
Write 0x4182 for enable access to CC32 with autoread

2.6 Zugriffs-Timeout

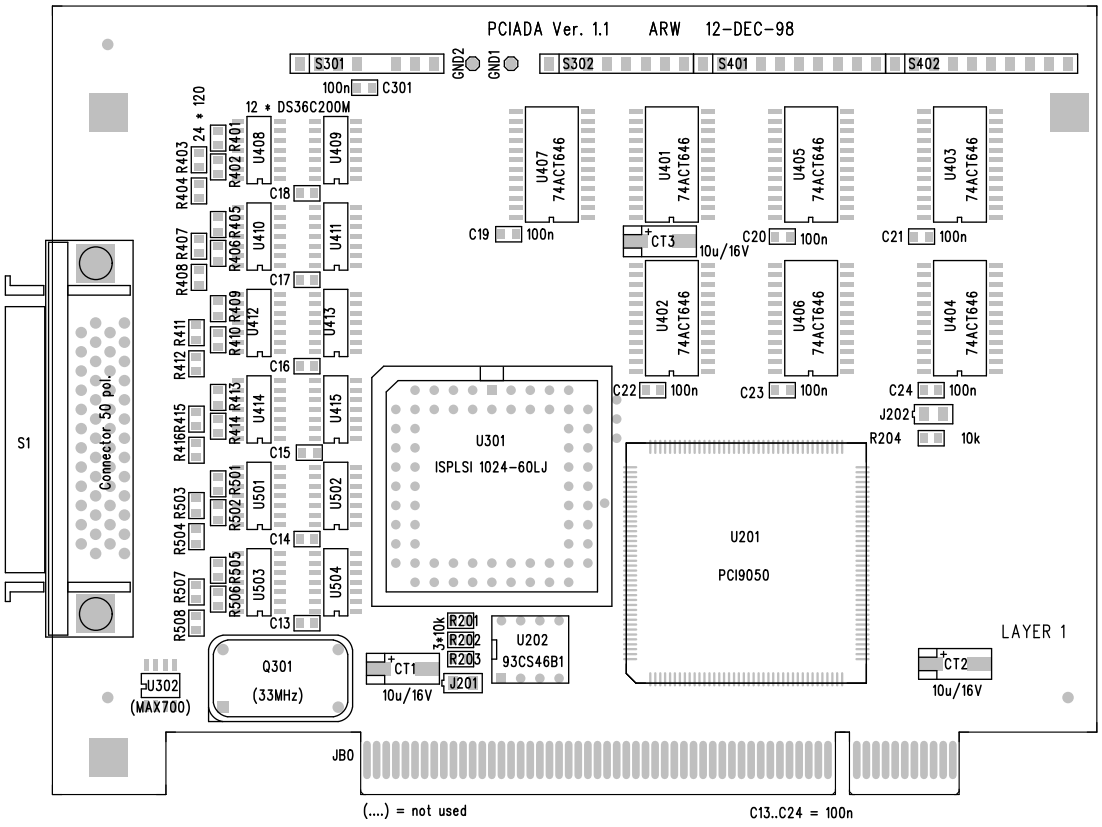
In PCIADA ist ein globaler Zugriffs-Timeout zum Absichern der Zugriffe zu CC32 realisiert. Beim Ablauf dieses Timeouts wird der laufende Zugriff ordnungsgemäß abgebrochen und der INTERRUPT2 ausgelöst. Dieser Timeout beendet den CC32-Zugriff bei abgeschaltetem Einschubrahmen oder nicht gestecktem Kabel.

Die Timeout Zeit ist fest auf ca. 35 usec eingestellt.

2.7 Stromaufnahme

Spannung	**Stromaufnahme**	**Leistung**
+5V	ca. 0,5 A	ca. 2,5W

2.8 Bestückungsplan PCIADA



3 CC32 CONTROLLER

3.1 Besonderheiten

3.1.1 FASTCAMAC basic Level 1

Beim Lesen mit der Funktion F = 5 liest der Controller im FASTCAMAC basic Level1 Mode die Daten von der selektierten Station, bis kein Q-Response mehr kommt.

Nach dem Lesen des ersten Datums liest der Controller selbständig das nächste Datum aus, und hält es für den nachfolgenden Rechner-Lesezyklus ohne Wartezeit bereit. Es wird mit der max. Lesegeschwindigkeit von PCIADA gelesen. Die Zeitersparnis liegt bei ca. 400 ns.

Wird der FASTCAMAC Level 1 Zyklus durch einen anderen F-Befehl unterbrochen, geht das bereits gelesene Datum verloren und der neue Befehl wird korrekt ausgeführt.

Funktionsänderung im FCL1_Mode!

Wenn beim FCL1-Mode **N**, **A oder F** sich ändert, wird der noch aktive Camac-Zyklus mit **S2** usw. beendet. Bereits gespeicherte Daten werden verworfen und ein neuer CAMAC- bzw. interner CC32-Zyklus wird ausgeführt.

Es ist darauf zu achten, dass bei eingeschaltetem AutoRead dieser abgeschaltet und wieder eingeschaltet werden muss, damit der CC32 den neuen NAF-Befehl erkennt.

3.1.2 CAMAC-Cycle-Tuning

Für jede CAMAC-Station kann die Zykluszeit von Beginn (BUSY = aktiv) bis zum S1 Befehl per Software gesteuert werden. Es sind optional 300 ns, 200 ns einstellbar. Bei der 200 ns Zeit kann die Pulbreite von S1 u. S2 optional auf 100 ns eingestellt werden.

3.1.3 ReadDoubleWord

Der CC32 kann mit **einem** Host-Readlong (32 bit read) auf dem CamacDataway **zwei** 16 bit Worte lesen. Dies ist bei jedem Read (auch bei FCL1) möglich. Hierzu muss im CC32 das **ReadDoubleWord-FlipFlop** (**RDW-FF**) gesetzt werden. Es ist darauf zu achten, dass bei eingeschaltetem AutoRead dieser abgeschaltet werden muss, wenn die NAF-Signale sich ändern.

Da im **RDW** Modus der Q-Status nicht mehr mitgelesen werden kann, wurde zusätzlich ein **QLAM-FF** und ein **QMASK-FF** eingebaut. Das QLAM-FF wird gesetzt, wenn die QMASK gesetzt ist und das Q-Signal mit der neg. Flanke von S1 inaktiv wird (wenn Q-LED ausgeht). Das QLAM-FF ist mit dem LAM-FF verodert (logisches OR).

Nun kann man durch pollen des Interruptregisters auf der PCIADA-Karte den LAM-Status abfragen oder einen Interrupt generieren.

3.1.4 DATAWAY-DISPLAY

Die an die CC32-NS (NS=Normalstation) Karte angeteckte CC32-LED Karte zeigt die Daten R1..R24 bzw. W1..W24, A1..A8, F1..F16 und. N1..N16 des letzten CAMAC-Transfers an. Ebenso die Signale Q, X, C u. Z.

Die Datenbits und einige Kontrollbits können mit der Funktion READ-LEDs gelesen werden, auch ohne gestecktes LED-Modul.

Mit Hilfe dieser Funktion können eventuelle Unterbrechungen bzw. Kurzschlüsse auf den Signalleitungen W1...W24 erkannt werden.

Die N-LEDs ändern sich auch bei lokalen CC32-Zugriffen.

3.1.5 Normal-Station von CC32

Zum Testen von CC32 wurden in die Normal-Station folgende Funktionen implementiert:

(Nn = CC32 Controlstation –1)

Write

Nn * A0 * F16 data <> 5	generiert Q u.X
Nn * A0 * F16 data = 5	generiert Q ,X u LAM (LAM 200ns aktiv)
Nn * A1 * F16 data = 0..15	Testzähler laden generiert Q u.X.

Read

Nn * A0 * F0 data = 0	generiert Q u.X

Read im FastCamacLevel1 Modus

Nn * A1 * F5 data = 0	decrement Testzähler
	generiert X u. Q nur wenn Inhalt Testzähler > 0

3.2 Berechnung von NAF

Die Adressbits A14..A2 werden beim Zugriff auf CC32 immer als **NAF-bits** interpretiert. Auch lokale Zugriffe sind als NAF-Befehle zu sehen.

Es sind nur Wort oder Longwort-Zugriffe auf CC32 möglich.

3.2.1 Wertigkeit von NAF

32K address CC32	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
CAMAC-Function-bit	N16	N8	N4	N2	N1	A4	A3	A2	A0	F8	F4	F2	F1	-	-

3.2.2 Berechnung von NAF

Die Wertigkeit von F16 ist bei der Berechnung von NAF nicht berücksichtigt!

Ein **WRITE** zu CC32 setzt selbsttätig **F16 = 0**.
Ein **READ** von CC32 setzt selbsttätig **F16 = 1**.

Der Adressoffset in das 32 kByte Speicherfenster wird wie folgt berechnet:

Beispiel in Pascal: **NAF := N shl 10 + A shl 6 + (F and $f)shl 2;**
Beispiel in C: **#define MAKE_CC32_OFFSET(N,A,F) ((N<<10) + (A<<6) + ((F & 0xf)<<2))**

Wenn Sie auf bestimmte CAMAC-Module sehr häufig und schnell zugreifen möchten, sollten Sie die aus NAF abgeleitete effektive Adresse als Konstante in Ihrer Software deklarieren. Sie sparen hierdurch Rechenzeit.

3.3 CC32-Addressmap

Dieser Adressbereich ist als 32k-Byte umfassender Raum in den PCI-Adressraum eingeblendet. Die Basisadresse im PCI-Adressraum wird dynamisch festgelegt. Bytezugriffe werden von CC32 nicht beantwortet, und werden durch einen TIME-OUT von PCIADA beendet. Alle Wort- und Langwortzugriffe werden von CC32 ohne Überprüfung beantwortet.

Bei Longword-Read von einer CC32-Wordaddrese ist D31..D16 gleich D15..D00.

Bei Longword-Write auf eine CC32-Wordaddrese wird nur D15..D00 übergeben.

Alle CC32-Zugriffe sind in der NAF-Notation beschrieben.

Der markierte Bereich kennzeichnet den Zugriff auf Stationen. Alle anderen Zugriffe werden für Sonderfunktionen im Zusammenhang mit dem CC32 verwendet.

Die Aufteilung ist wie folgt:

NAF	Access	WR-Function / F16-bit=1	RD-Function / F16-bit=0
N31*A0*Fx	Word	**CC32 RESET**	-
N30*A2*Fx	Word	**CYCLE-TUNE-HIGH** D15..D00 >> N24..N17	**CYCLE-TUNE-HIGH** D15..D00 << N24..N17
N30*A1*Fx	Word	**CYCLE-TUNE-MID** D15..D00 >> N16-N9	**CYCLE-TUNE-MID** D15..D00 << N16..N9
N30*A0*Fx	Word	**CYCLE-TUNE-LOW** D15..D00 >> N8..N1	**CYCLE-TUNE-LOW** D15..D00 << N8..N1
N29*A0*Fx	Lword	-	**LED-Status** D23..D00 << LED24..LED1 D27..D24 << C,Z,CT1,CT0 D31..D28 << Q,X,INH,LAM-FF
N28*A4*Fx	Lword	-	**LAM-BUS** D23..D00 << LAM24..LAM1 D31..D24 is equal LAM-MASK
N28*A3*Fx	Lword	-	**LAM-NOT = LAMn & !LMASKn** D23..D00 << NOT24..NOT1 D31..D24 is equal LAM-MASK
N28*A2*Fx	Lword	-	**LAM-AND = LAMn & LMASKn** D23..D00 << AND24..AND1 D31..D24 is equal LAM-MASK
N28*A1*Fx	Lword	**LAM-MASK** D23..D00 >> LMASK23-LMASK0 D24 = QMASK-FF /1=on / 0=off D31..D24 = xx	**LAM-MASK** D23..D00 << LMASK24..LMASK1 D24 = QMASK-FF D25 = QLAM-FF D27,D26 = 0 D28 = LAM-BUS-OR D29 = LAM-NOT-OR D30 = LAM-AND-OR D31 = LAM-FF
N28*A0*Fx	Word	**LAM_FF and QLAM-FF reset** D15..D00 = xx	**LAM-FF Status** D00 = LAM-FF 0n = 1 D01 = 1 QLAM-FF on = 1 D15..D02 = 0
N27*A0*Fx **N27*A1*Fx** **N27*A2*Fx** **N27*A3*Fx**	Word	**INHIBIT on** **INHIBIT off** **RDW-FF on** **RDW-FF off** D31..D00 = xx	**INHIBIT Status** D00 = INHIBIT on = 1 D01 = INHIBIT Dataway on = 0 D02 = RDW-FF on = 1 D15..D03 = 0
N26*A0*Fx	Lword	**Broadcast-MASK** D23..D00 >> BMASK24..BMASK1	**Broadcast-MASK** D23..D00 << BMASK24..BMASK1 D24..D31 << 0

N25*Ax*Fx	Lword	**Broadcast-WR=allN & BMASKn** D23..D00 >>W1..W24	-
N1-24*Ax*Fx	Lword	**CAMAC-DATAWAY WRITE *1** D23..D00 >> W24..W1 *3	**CAMAC-DATAWAY READ *1** D23..D00 << R23..R00 D29..D24 = 0 D31,D30 Q,X
N1-24*Ax*Fx	Word	**CAMAC-DATAWAY WRITE *1** D00..D15 > W1..W16 *3	**CAMAC-DATAWAY READ *1** D00..D15 < R1..R16 *4
N0*A0*Fx **N0*A1*Fx** **N0*A2*Fx** **N0*A3*Fx**	Word	**CAMAC C *2** **CAMAC Z *2** **CAMAC C + INHIBIT off *2** **CAMAC Z + INHIBIT on *2** D15..D00 = xx	**CC32-STATUS** D03..D00 << Q,X,INH,LAM-FF D07..D04 << Modul-Number D11..D08 << FPGA-Revision D15..D12 << Modul-Type 1000b

*1 Standard CAMAC -Access
*2 Standard CAMAC -Access without S1
*3 no W-Data on CAMAC -Dataway when F8-bit is active
*4 if test Q- or X-Status then use Lword-Access

3.4 CC32-Status

N0*A0*Fx (Read Word)

Aus diesem Register kann die Einstellung des CC32 erfragt werden.

CC32-Status (word read access only)

Bit		RD	WR	after Init
15..12	Module type identification, 1000b for CC32 (0001b VMEMM)	yes	no	1000b
11..8	FPGA-Revision3 / Nov-01	yes	no	0011b
7..4	Module number, Coding of Jumpers J304..J301	yes	no	Jumpers
3	Q – Response	yes	no	x
2	X – Response	yes	no	x
1	State of Inhibit-Flip-Flop	yes	no	0
0	State of LAM-Flip-Flop	yes	no	0

Hinweis: Mit Hilfe der Modul-Identifikation und der Modul-Nummer kann das angeschlossene Interface identifiziert werden.

3.5 CC32-C,Z,Inhibit,LAM-FF

N0*A0*Fx	=	Camac Clear	(Write Word)
N0*A1*Fx	=	Camac Initialize	(Write Word)
N0*A2*Fx	=	Camac Clear + Inhibit reset	(Write Word)
N0*A3*Fx	=	Camac Clear + Inhibit set	(Write Word)
N27*A2*Fx	=	**R**ead**D**ouble**W**ord on	(Write Word)
N27*A3*Fx	=	**R**ead**D**ouble**W**ord off	(Write Word)
N27*A0*Fx	=	Inhibit set	(Write Word)
N27*A1*Fx	=	Inhibit reset	(Write Word)
N28*A0*Fx	=	LAM-FF reset	(Write Word)

3.6 Broadcast-Mask-Register Write/Read

N26*A0*Fx (Write Lword)

Alle N-Stationen für die das Broadcast Mask-bit = 1 gesetzt ist werden aktiviert.

Data	D23	D22	D21..D3	D2	D1	D0
Broadcast-Mask for:	N24	N23	N22..N4	N3	N2	N1

3.7 Broadcast-Write

N25*Ax*Fx (Write Word or Lword)

Standart Camac-Zycklus mit Ax und Fx

N1..N24 -wird aktiv wenn Broadcast-Mask-Bit gesetzt ist

3.8 LAM-Mask-Register Write/Read

N28*A1*Fx (Write Lword)

Alle N-Stationen für die das Broadcast Mask-bit = 1 gesetzt ist koennen das LAM-FF aktiviert.

Das QLAM-FF wird gesetzt, wenn die QMASK gesetzt ist und das Q-Signal mit der neg. Flanke von S1 inaktiv wird. (wenn Q-LED ausgeht) Das QLAM-FF ist mit dem LAM-FF verodert (logisches OR).

Data	D24	D23	D22	D21..D3	D2	D1	D0
Enable LAM from Station	QLAM-FF	N24	N23	N22..N4	N3	N2	N1

Ein neg. LAM-Flanke wird nur an das LAM-Flip-Flop gegeben, wenn das zugehörige LAM-Mask-bit gleich 1 ist. Das LAM-FF bleibt solange gesetzt bis es mit N28*A0*F16 zurückgesetzt wird.

Beim Lesen werden die Statusbits

D25 = 1 QLAM-FF is set (LAM aktiv)

D28 = 1 (LAM-BUS-OR) wenn ein oder mehrere LAMs von allen Stationen aktiv sind.

D29 = 1 (LAM-NOT-OR) wenn ein oder mehrere LAMs der Stationen mit LAM-Maskbit =0 aktiv sind.

D30 = 1 (LAM-AND-OR) wenn ein oder mehrere LAMs der Stationen mit LAM-Maskbit = 1aktiv sind.

D31 = 1 (LAM-FF) wenn gesetzt.

3.9 LAM-AND-Status Read

N28*A2*Fx (Read Lword)

Dxx = 1 wenn LAM = aktiv und LAM-Maskbit = 1 ist.

Data	D23	D22	D21..D3	D2	D1	D0
LAM status & LMASK	N24	N23	N22..N4	N3	N2	N1

Inhalt von D28..D31 wie bei **3.8**

3.10 LAM-NOT-Status Read

N28*A3*Fx (Read Lword)

Dxx = 1 wenn LAM = aktiv und LAM-Maskbit = 0 ist.

Data	D23	D22	D21..D3	D2	D1	D0
LAM Status & not LMASK	N24	N23	N22..N4	N3	N2	N1

Inhalt von D28..D31 wie bei **3.8**

3.11 LAM-BUS-Status Read

N28*A4*Fx (Read Lword)

Dxx = 1 wenn LAM = aktiv ist.

LAM-BUS	D23	D22	D21..D3	D2	D1	D0
LAM Status CAMAC-Bus	N24	N23	N22..N4	N3	N2	N1

3.12 LED-Status Read

N29*A0*Fx (Read Lword)

Die LED24..LED1 entsrechen W-Data bzw. R-Data vom letzten Camac Zykluss

Data	D23	D22	D21..D3	D2	D1	D0
LED-Status	LED24	LED23	LED22..LED3	LED3	LED2	LED1

Data	D31	D30	D29	D28	D27	D26	D25	D24
LED-Status	Q	X	Inhibit	LAM-FF	CT1	CT0	Z	C

3.13 CAMAC Cycle-Tune-Register Write/Read

N30*A2*Fx High-Register fuer Station N24..N17 (Write/Rread Word

N30*A1*Fx Mid-Register fuer Station N16..N9 (Write/Rread Word)

N30*A0*Fx Low-Register fuer Station N8..N1 (Write/Rread Word)

Die Zeit von Beginn des CAMAC-Zyklus bis zur neg. S1-Flanke kann individuell für jede Station auf 300 bzw. 200 ns eingestellt werden. Ebenso kann die Pulsbreite von S1 u. S2 auf 100 ns verkürzt werden.

Pro Station werden hierfür 2-bit CT1 u. CT0 (nachfolgend mit Nx-1 u. Nx-0 bez.) benötigt.

Nx-1,Nx-0 = 00 > 400 ns CAMAC-Standart

Nx-1,Nx-0 = 01 > 300 ns

Nx-1,Nx-0 = 10 > 200 ns

Nx-1,Nx-0 = 11 > 200 ns / S1 u. S2 = 100ns

Registermap:

DATA	D07	D06	D05	D04	D04	D02	D01	D00
Cycle-tune high	N20-1	N20-0	N19-1	N18-0	N18-1	N18-0	N17-1	N16-0
Cycle-tune mid	N12-1	N12-0	N19-1	N11-0	N18-1	N10-0	N9-1	N9-0
Cycle-tune low	N4-1	N4-0	N3-1	N3-0	N2-1	N2-0	N1-1	N1-0

DATA	D15	D14	D13	D12	D11	D10	D09	D08
Cycle-tune high	N24-1	N24-0	N23-1	N23-0	N22-1	N22-0	N21-1	N21-0
Cycle-tune mid	N17-1	N16-0	N^5-1	N15-0	N14-1	N14-0	N13-1	N13-0
Cycle-tune low	N8-1	N8-0	N7-1	N7-0	N6-1	N6-0	N5-1	N5-0

Achtung: Diese Optionen entsprechen nicht der CAMAC Spezifikation !!!

Ob ein CAMAC-Module diese Anforderungen erfüllt, muss vom Anwender durch Test ermittelt werden.

3.14 CC32-Reset

N31*A0*Fx **(**Write Word)

Folgende Register werden zurückgesetzt:

Inhibit, LAM-FF,

BROADCAST-MASK-, CYCLE-TUNE- und LAM-MASK-REGISTER

3.15 Interrupts

Der von PCIADA generierte Timeout Interrupt sollte von der Software auf den Interrupt Vektor Nummer 1 gespiegelt werden.

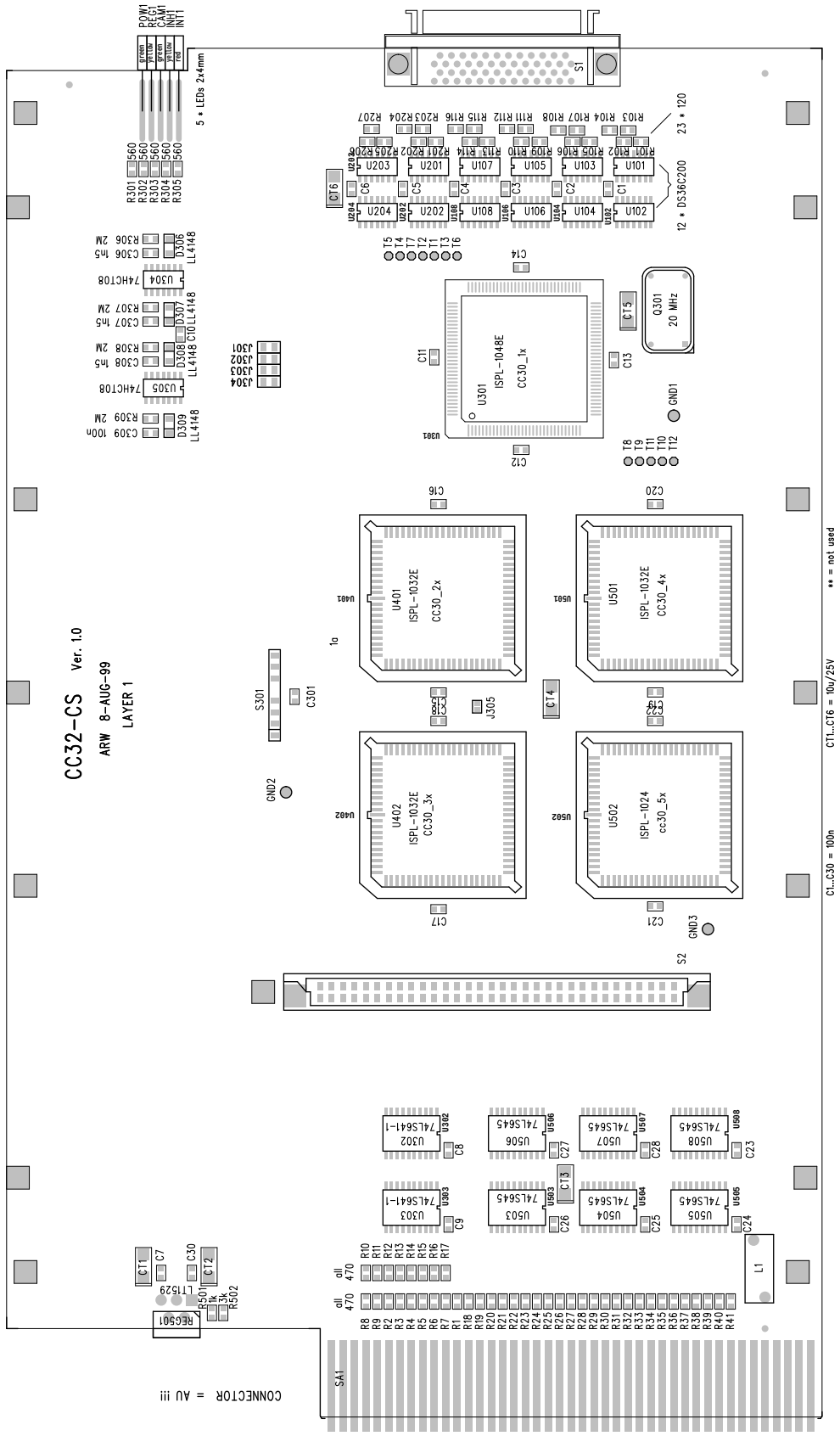
Der von CC32 generierte Interrupt sollte von der Software auf den Interrupt Vektor Nummer 2 gespiegelt werden.

Interrupt Ursache	**Vektor Nr.**
PCIADA erzeugter Interrupt (Timeout)	1
LAM-FF erzeugter Interrupt	2

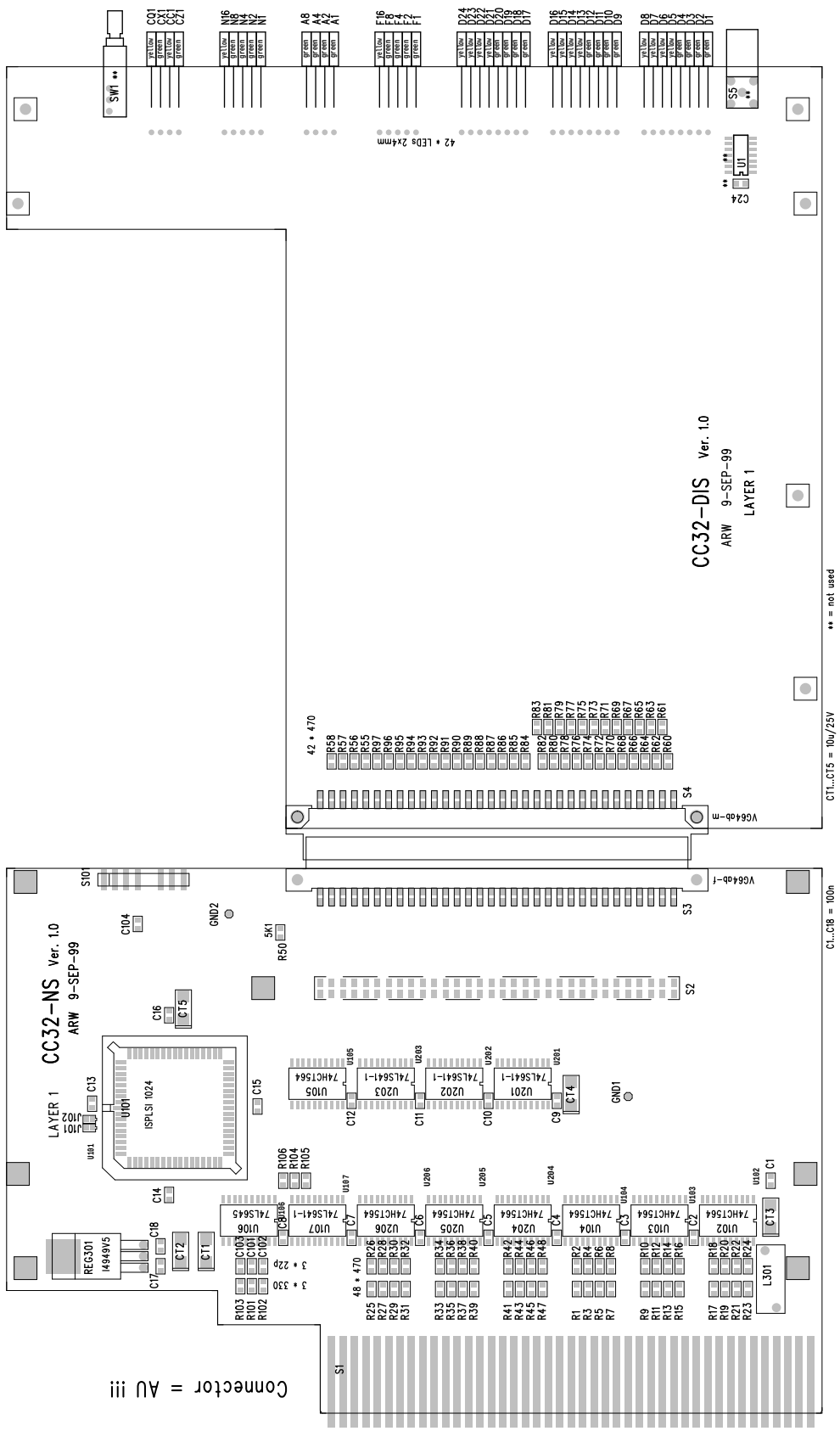
3.16 Stromaufnahme

Spannung	**Stromaufnahme**	**Leistung**
+6V	1,7 A	10,2 W

3.17 Bestückungsplan CC32 Control-Station



3.18 Bestückungsplan CC32 Normal-Station



3.19 Camac Steckerbelegung

CC32 Bus-Belegung

Normal-Station

Sig.Top	Nr.	Sig.Bott.
	1	B
	2	F16
	3	F8
	4	F4
	5	F2
X	6	F1
I	7	A8
C	8	A4
N	9	A2
L	10	A1
S1	11	Z
S2	12	Q
W24	13	W23
W22	14	W21
W20	15	W19
W18	16	W17
W16	17	W15
W14	18	W13
W12	19	W11
W10	20	W9
W8	21	W7
W6	22	W5
W4	23	W3
W2	24	W1
R24	25	R23
R22	26	R21
R20	27	R19
R18	28	R17
R16	29	R15
R14	30	R13
R12	31	R11
R10	32	R9
R8	33	R7
R6	34	R5
R4	35	R3
R2	36	R1
	37	
	38	
	39	
	40	
	41	
	42	+6
Gnd	43	Gnd

Contol-Station

Sig.Top	Nr.	Sig.Bott.
	1	B
	2	F16
	3	F8
	4	F4
	5	F2
X	6	F1
I	7	A8
C	8	A4
	9	A2
	10	A1
S1	11	Z
S2	12	Q
L24	13	N24
L23	14	N23
L22	15	N22
L21	16	N21
L20	17	N20
L19	18	N19
L18	19	N18
L17	20	N17
L16	21	N16
L15	22	N15
L14	23	N14
L13	24	N13
L12	25	N12
L11	26	N11
L10	27	N10
L9	28	N9
L8	29	N8
L7	30	N7
L6	31	N6
L5	32	N5
L4	33	N4
L3	34	N3
L2	35	N2
L1	36	N1
	37	
	38	
	39	
	40	
	41	
	42	+6
Gnd	43	Gnd