grifo® Mini Module Test 2

Inbetriebnahme der Module GMM AM08 und GMM AM32

1. Vorbemerkung

GMM TST 2 (grifo® Mini Module Test 2) ist ein Experimentierboard für die folgenden grifo® Mini Module:

- GMM AM32 Mini Modul mit Atmel ATmega32L
- GMM AM08 Mini Modul mit Atmel ATmega8L
- GMM AC2 Mini Modul mit Atmel T89C51AC2
- GMM 5115, Mini Modul mit Atmel T89C5115
- GMM 932 Mini Modul mit Philips P89LPC932

Im folgenden wird die Handhabung und Inbetriebnahme der Module GMM AM32 und GMM AM08 unter BASCOM-AVR betrachtet.

2. grifo® Mini Module Test 2

Das Experimentierboard (Abbildung 1) weist die folgenden Merkmale auf:

- zwei DIL40 Sockel (Z1, Z2) zur Aufnahme der o.a. grifo® Mini Module
- Standard 2.1 mm Stecker zur Spannungsversorgung CN1
- Betriebsspannung 7 – 12V AC oder 9 – 16 V DC
- Anzeige der (intern erzeugten) +5 V DC durch gelbe LED L1
- RESET Taster
- Buzzer (durch I/O Leitung angesteuert)
- DB9 Steckverbinder (female) CN5 für RS-232
- 10-Pin Steckverbinder CN7 für Atmel AVR ISP programmer
- DB9 Steckverbinder (female) CN6 zur ISP über RS-232 und PonyProg
- Zwei 20-Pin Steckverbinder für I/O Leitungen der Mini Module
- LCD 20x2 mit einstellbarer Hintergrundbeleuchtung

- Matrix-Keyboard mit 16 Tasten
- Zwei Taster und zwei LEDs
- Jumper für AD-Referenzspannung
- Abmessungen 100 x 212 x 30 mm.

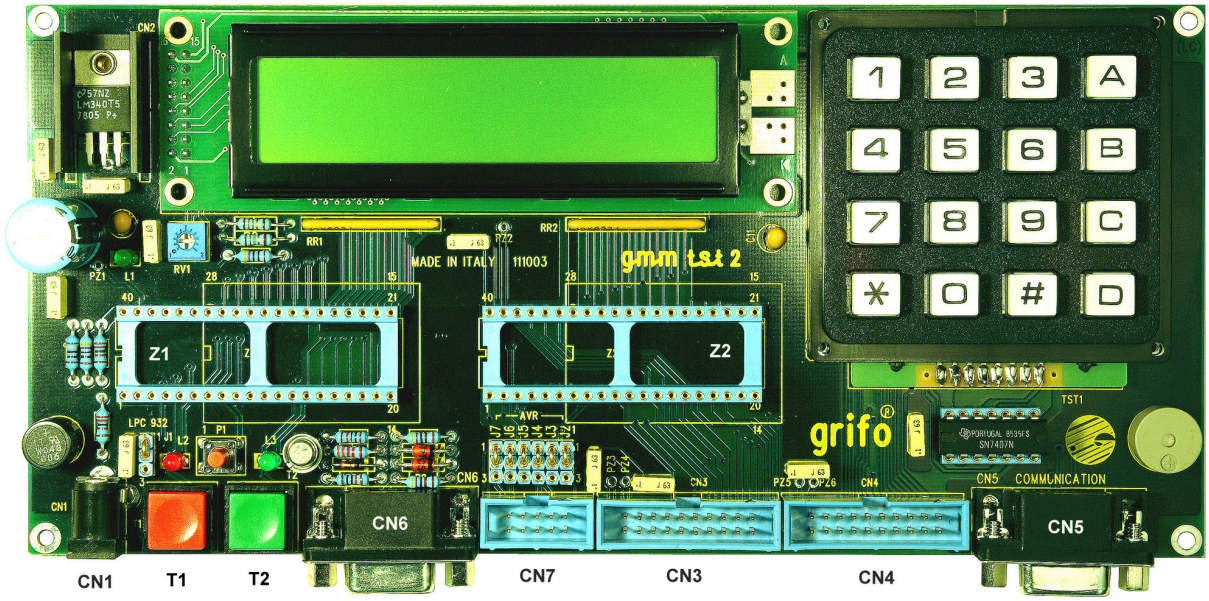

Abbildung 1 GMM TST2 - Komponentenseite

2.1. Steckverbinder

2.1.1. CN1 – Spannungsversorgung

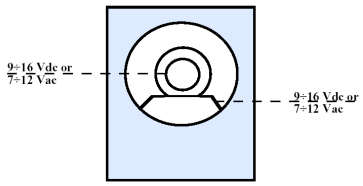

Abbildung 2 Steckerbelegung CN1

2.1.2. CN5 - RS 232 Kommunikationsinterface

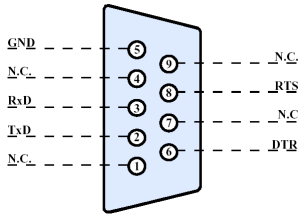

Abbildung 3 Steckerbelegung CN5

2.1.3. CN3 – I/O

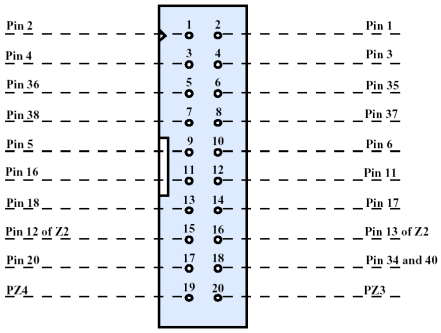

Abbildung 4 Steckerbelegung CN3

Die Zuordnung der I/O Leitungen zu den I/O Pins der Mini Module ist der folgenden Tabelle zu entnehmen.

Pin DIP40 (Z1, Z2)	*GMM AM08*	*GMM AM32*
Pin1	-	PA4
Pin2	-	PA5
Pin3	-	PC2
Pin4	-	PC3
Pin5	-	PD4
Pin6	-	DSW1.8
Pin11	N.C	/INT
Pin12 (Z2)	PC5	PC0
Pin13 (Z2)	PC4	PC1
Pin16	ADC6	PA6
Pin17	N.C.	PD7
Pin18	PB5	PB7
Pin20	GND	GND
Pin34	+5 V DC	+5 V DC
Pin35	-	PC4
Pin36	-	PC5
Pin37	-	PC6
Pin38	-	PC7
Pin40	-	+5 V DC

2.1.4. CN4 – I/O

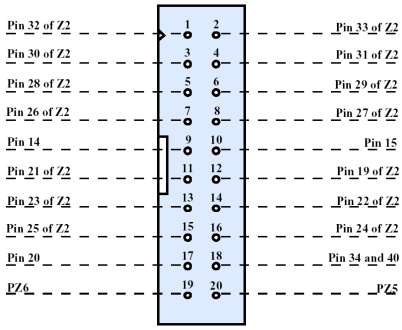

Abbildung 5 Steckerbelegung CN4

Die Zuordnung der I/O Leitungen zu den I/O Pins der Mini Module ist der folgenden Tabelle zu entnehmen.

Pin DIP40 (Z2)	*GMM AM08*	*GMM AM32*
Pin14	PB3	PB5
Pin15	PB4	PB6
Pin19	PC3	PA3
Pin21	PC2	PA2
Pin22	PD5	PB1
Pin23	PD4	PB0
Pin24	PD3	PD3
Pin25	PD2	PD2
Pin26	PD7	PB3
Pin27	PD6	PB2
Pin28	PB2	PB4
Pin29	PB0	PD6
Pin30	PB1	PD5
Pin31	PC1	PA1
Pin32	PC0	PA0
Pin33	ADC7	PA7

2.1.5. CN6 – RS-232 serieller ISP PONYPROG Programmierstecker

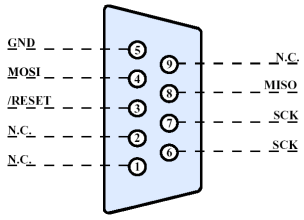

Abbildung 6 Steckerbelegung CN6

2.1.6. CN7 - AVR ISP Programmierstecker

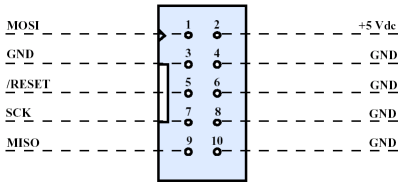

Abbildung 7 Steckerbelegung CN7

2.2. Buzzer

Ein selbst-schwingender Buzzer erzeugt einen kontinuierlichen Ton von etwa 1 kHz, wenn er durch durch einen Mikrocontroller angesteuert wird. Der Buzzer ist an Pin15 von Z1 angeschlossen.

2.3. Taster und LEDs

Zwei Taster (T1, T2) und zwei LEDs (L2, L3) mit Vorwiderständen sind mit den Pin12 (T1, L2) und Pin13 (T2,L3) von Z1 verbunden (Abbildung 8).

Eine Tastenbetätigung zieht den Eingang nach GND. Die LED wird durch Lo am Ausgang eingeschaltet.

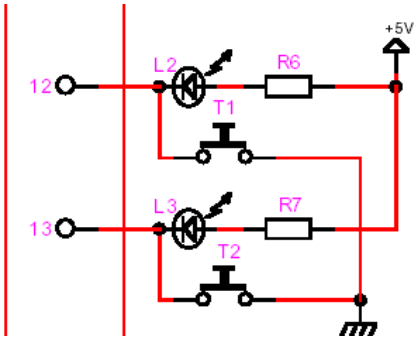

Abbildung 8 Taster und LEDs an Z1

2.4. LCD und Matrix-Keyboard

LCD und Matrix-Keyboard sind unabhängig voneinander an Z1 geschaltet (Abbildung 9).

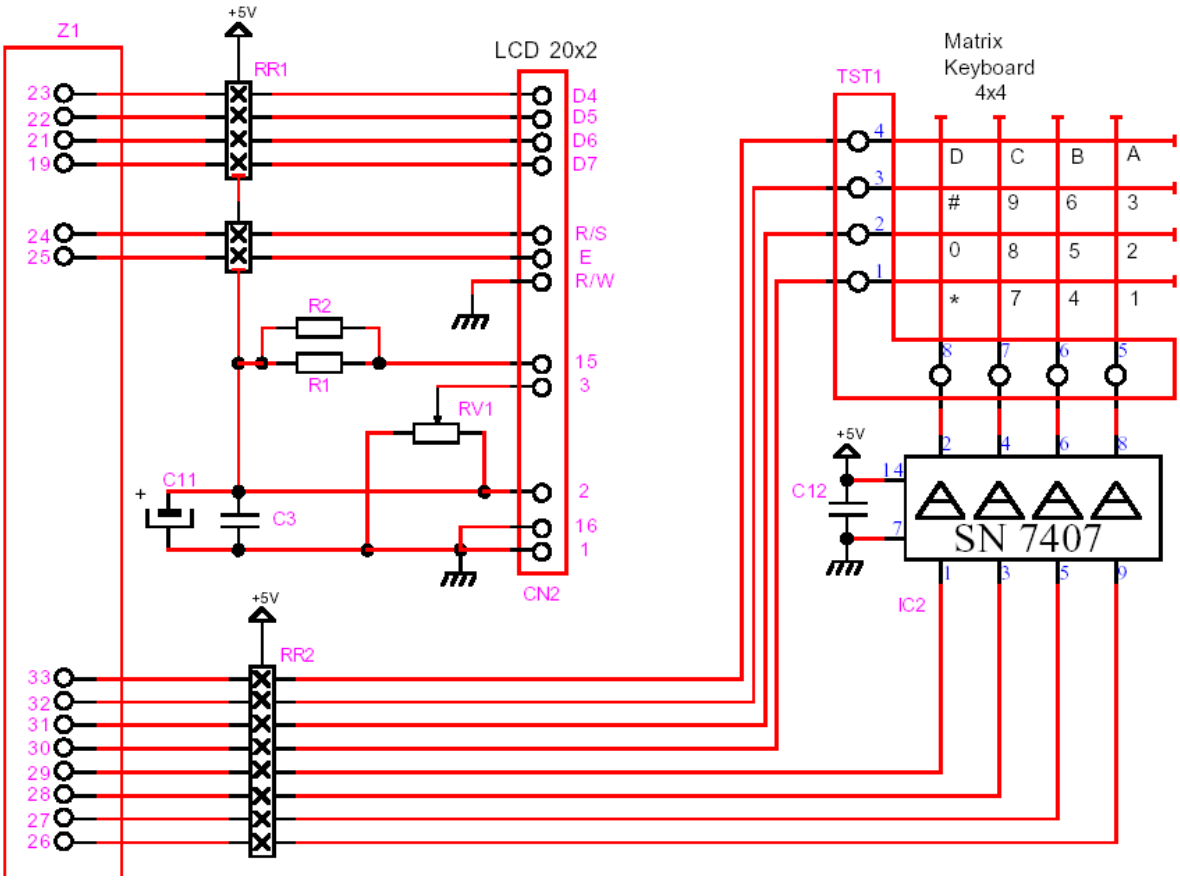

Abbildung 9 LCD und Matrix-Keyboard an ZI

2.5. Jumper

Eine Reihe von Jumpern dienen der Konfiguration des GMM TST2. Die erforderlichen Jumperstellungen bei Programmierung mit PonyProg sind durch * markiert.

Jumper	*Position*	*Zweck*	*ISP mit PonyProg*
1	1-2 2-3	Verbindet Pin7 (Z1, Z2) mit RTS Verbindet Pin7 (Z1, Z2) mit + 2,5 V DC (Referenzspannung)	
2	1-2 2-3	Verbindet Pin15 (Z1) mit MISO Verbindet Pin15 (Z1) mit Buzzer	*
3	1-2 2-3	Verbindet Pin8 (Z1) mit /RESET Verbindet Pin8 (Z1) mit DTR	*
4	1-2 2-3	Enables /RESET von PonyProg Anschluss CN6 Enables /RESET von ISP Anschluss CN7	*
5	1-2 2-3	Enables MOSI von PonyProg Anschluss CN6 Enables MOSI von ISP Anschluss CN7	*
6	1-2 2-3	Enables SCK von PonyProg Anschluss CN6 Enables SCK von ISP Anschluss CN7	*
7	1-2 2-3	Enables MISO von PonyProg Anschluss CN6 Enables MISO von ISP Anschluss CN7	*

3. grifo® Mini Module

3.1. GMM AM08

Das Mini Modul GMM AM08 eist folgende Eigenschaften aus:

- Abmessungen 20.7 x 38.7 x 12.8 mm
- 28-polige Stiftleiste passend in EPROM Sockel
- Spannungsversorgung +3.0 V DC bis +5 V DC
- 7.3728 MHz Quarz
- 8 KB Flash Memory Programmspeicher, bis zu 1 KB Flash memory für optionalen Bootloader, 512 Bytes EEPROM, 1024 Bytes RAM Datenspeicher
- Idle und Power Down Mode verfügbar
- AD-Umsetzer mit sechs Kanälen zu 10 Bit und zwei kanälen zu 8 Bit Auflösung, Umsetzzeit 5 µs
- 18 Interrupts
- programmierbarer Watchdog
- 23 I/O herausgeführt
- Hardware-UART mit bis zu 115 kBaud, TTL oder RS-232 Pegel selektierbar
- Synchrone serielles Interfaces SPI und I^2C
- Resetschaltung
- Status-LED
- Progammierung durch ISP

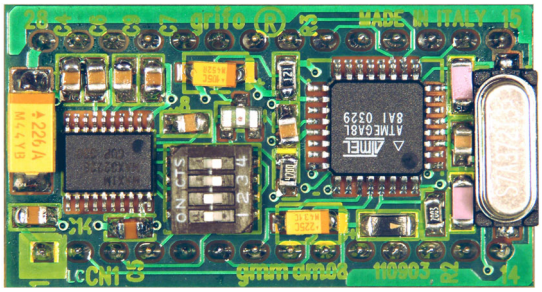

Abbildung 10 GMM AM08 Bestückungsseite

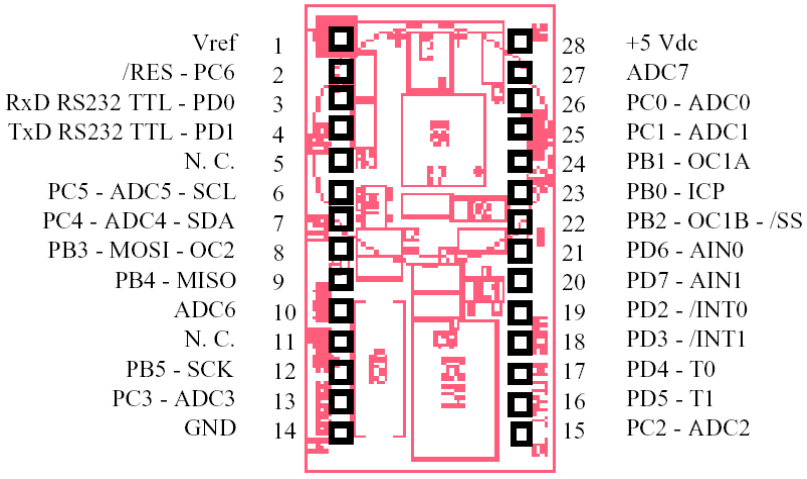

Abbildung 11 GMM AM08 Pinbelegung

3.2. GMM AM32

Das Mini Modul GMM AM08 eist folgende Eigenschaften aus:

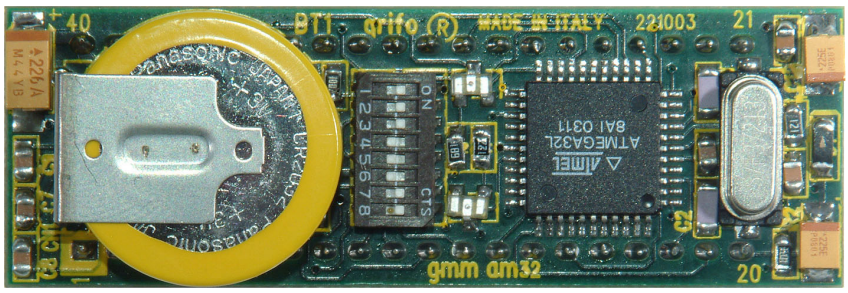

Abbildung 12 GMM AM32 Bestückungsseite

4. Programmierung

Zur Programmierung der Module GMM AM32 bzw. GMM AM08 setzt man am einfachsten den Programmer PonyProg ein.

PonyProg ist eine von Claudio Lanconelli [http://www.lancos.com/prog.html] entwickelte Software, welche unter jeder Version von Windows läuft und unterschiedliche Mikrocontroller über die serielle Schnittstelle programmieren kann. (Eine Reihe von Parallelport-Programmern machen unter Windows2000 u.ä. Schwierigkeiten.)

Der Anschluss CN6 ist mit einem COM-Port des Entwicklungs-PCs zu verbinden und das Programm PonyProg zu starten. Die erforderliche Hardware ist auf dem GMM TST 2 enthalten.

Die folgenden Screenshots zeigen die erforderlichen Schritte zum Programmieren eines AVR Mikrocontrollers am Beispiel eines GMM AMM32.

PonyProgr wird gemäss Abbildung 13 auf die serielle Schnittstelle eingestellt.

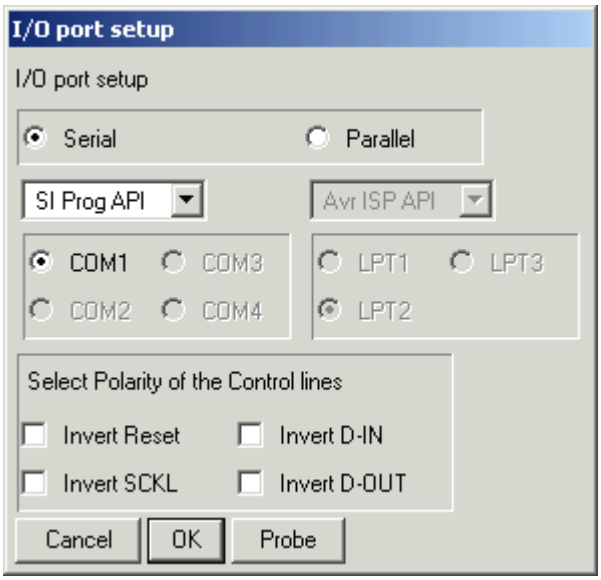

Abbildung 13 PonyProg SetUp

Die Auswahl des zu programmierenden Bausteins erfolgt gemäss Abbildung 14.

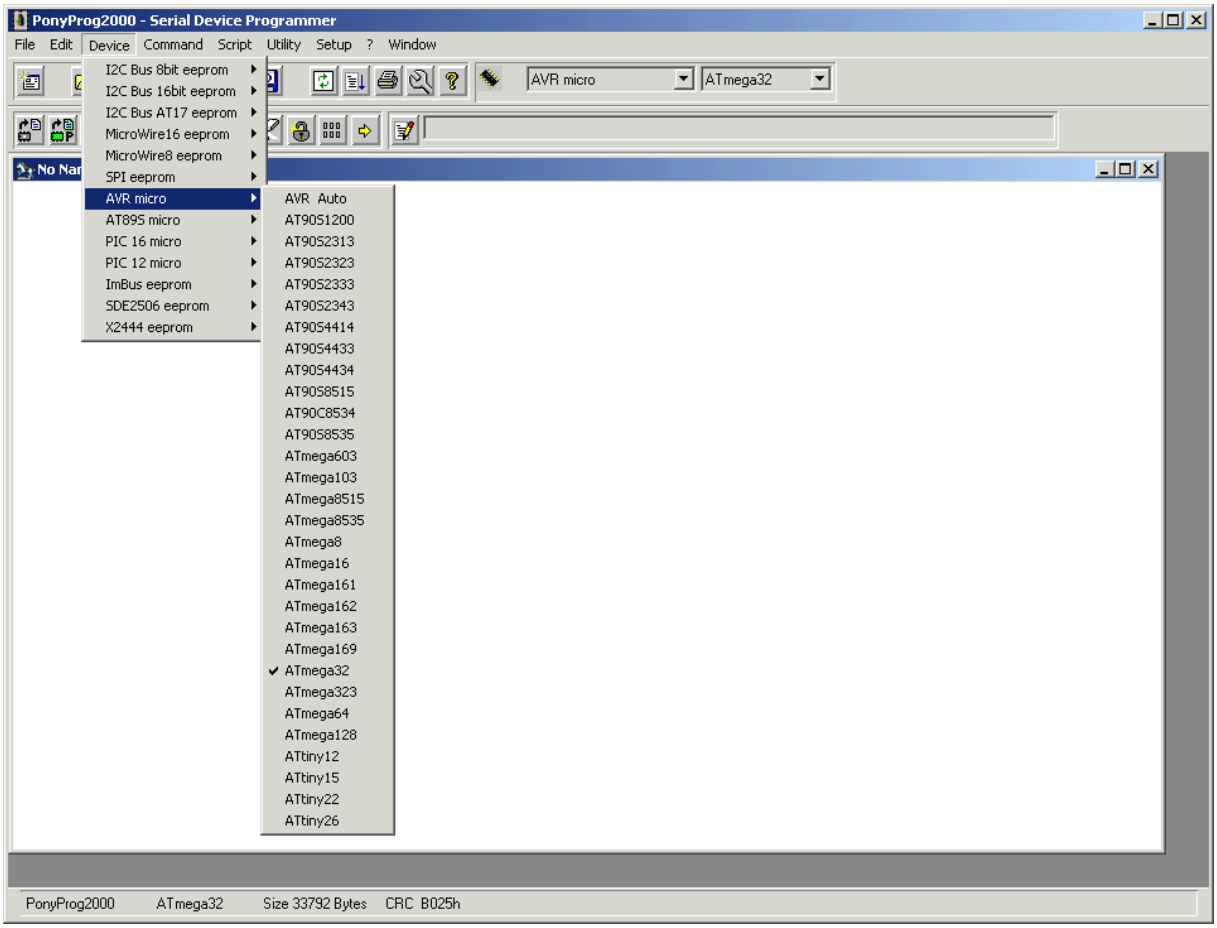

Abbildung 14 Auswahl des zu programmierenden Mikrocontrollers

Die Fuse und Lock Bits können gemäss Abbildung 15 gelesen und geschrieben werden. Hier sind die Einstellungen des Auslieferungszustands gelesen worden. Bevor die Fuse und Lock Bits verändert werden, muss unbedingt das Datenblatt konsultiert werden.

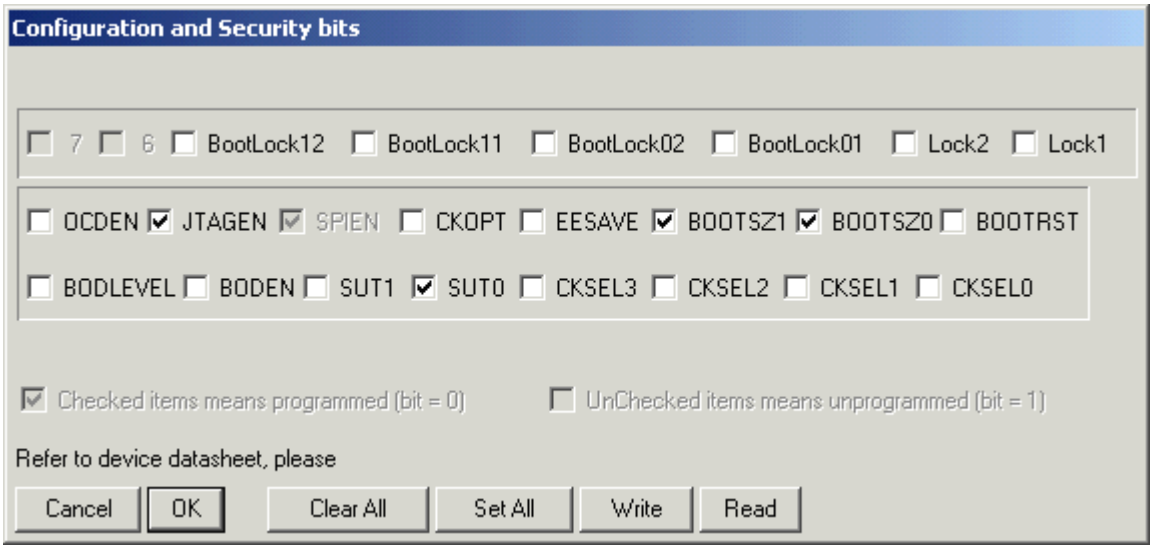

Abbildung 15 Lesen/Schreiben der Fuse und Lock Bits

Der GMM AM32 wird mit einem Demoprogramm programmiert ausgeliefert. Mit einem beliebigen Terminalprogramm kann die Datenausgabe an CN5 verfolgt werden. Abbildung 16 zeigt die Ausgaben und die Schnittstellenparameter in der Statuszeile.

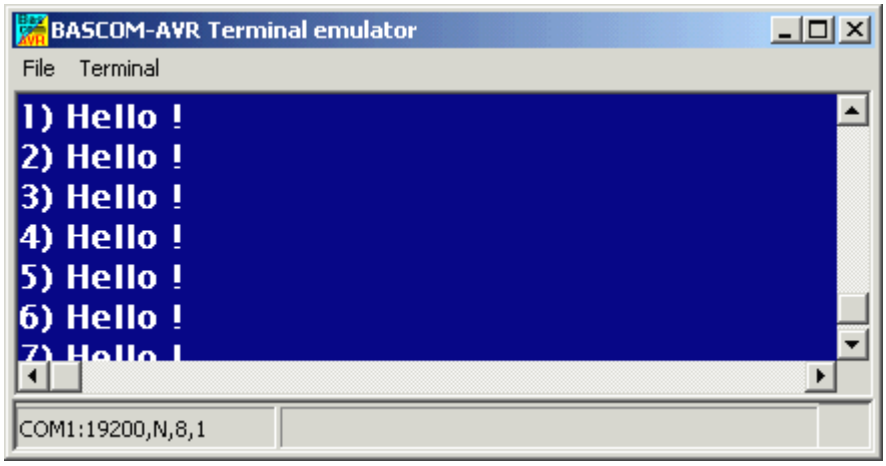

Abbildung 16 Ausgaben des Demoprogramms

Das Programm selbst kann ausgelesen und analysiert werden. Gleichermassen kann ein Hexfile geladen und programmiert werden. Abbildung 17 zeigt ein geladenes Programm als Memorydump. Das GMM AM32 kann nun programmiert werden.

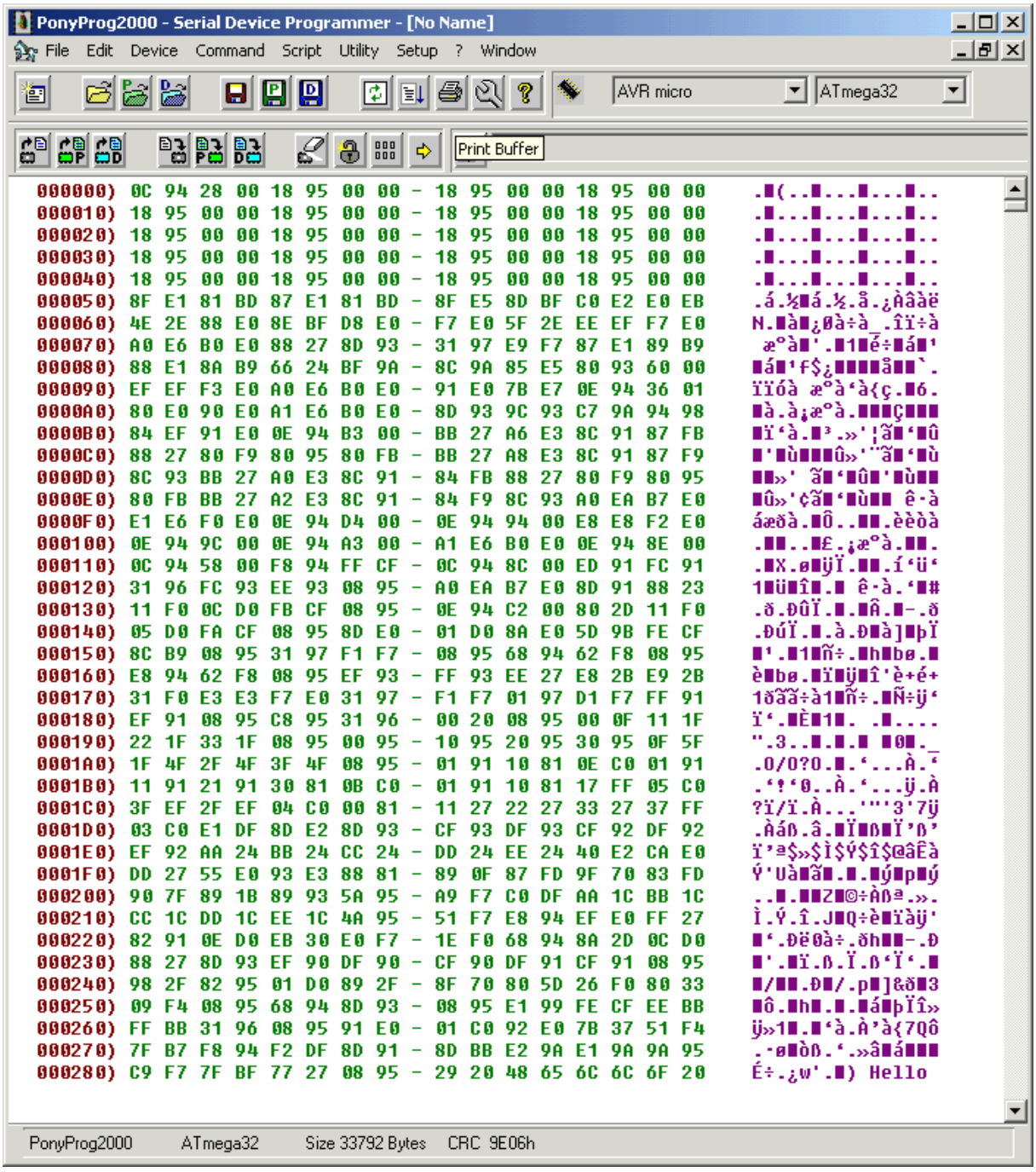

Abbildung 17 Memory Dump des Demoprogramms

Der PonyProg Programmer kann gemäss Abbildung 18 mit BASCOM-AVR verknüpft oder eigenständig gestartet werden.

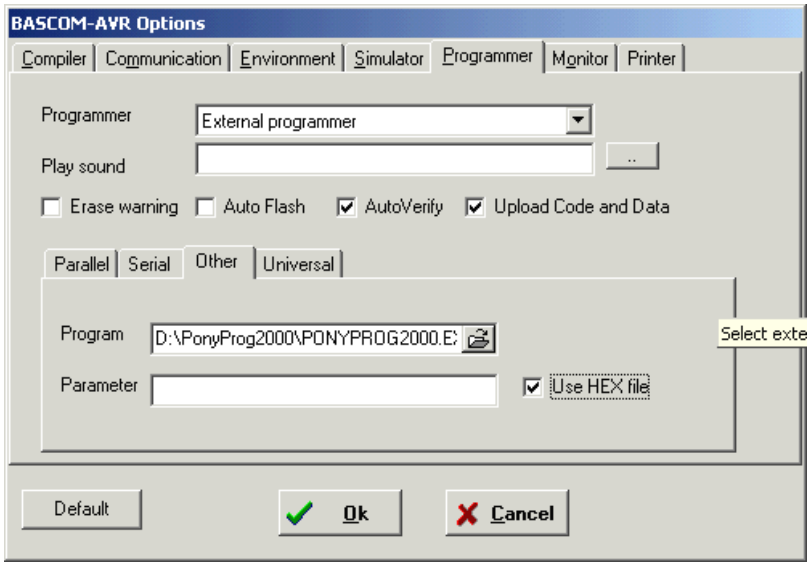

Abbildung 18 PonyProg als externer Programmer

5. Programmentwicklung mit BASCOM-AVR

Der erste Test für eine neues Mikrocontrollermodul ist meist ein Programm der "Hello World"-Klasse. Listing 1 zeigt ein solches, schon etwas erweitertes Beispiel.

Ein Timerinterrupt inkrementiert die Systemsekunde, die dann in einer Endlosschleife gemeinsam mit dem Text " Hello World - this is GMM AM32 made by grifo " über RS-232 ausgegeben wird. Die serielle Ausgabe erfolgt mit 19200 Baud. Abbildung 19 zeigt die Ausgaben im Terminalprogramm.

Abbildung 19 Hyperterminal Mitschnitt

```
'-----------------------------------------------------------
' HELLO WORLD
' First test of grifo Mini Modul GMM AM32
'-----------------------------------------------------------
$regfile = "m32def.dat"
$crystal = 7372800
$baud = 19200

Config Clock = User

On Compare1a Timerinterrupt             ' interrupt vector for output compare interrupt

Dim Lsyssec As Long
Dim Lsyssec_old As Long
Dim Strtime As String * 8
```

```
Dim W1 As Word                          ' Workaround to get _DIV16
W1 = W1 / W1

' Timer1 Configuration                  ' initializes sec timer
Tccr1a = 0
Tccr1b = &H0D

Ocr1ah = &H38
Ocr1al = &H40

Enable Compare1a
Enable Interrupts

Do
  Do
    Lsyssec = Syssec()
  Loop Until Lsyssec <> Lsyssec_old
  Print "Hello World - this is GMM AM32 made by grifo"
  Strtime = Time(lsyssec)               ' convert to time string
  Print "Running " ; Strtime
  Lsyssec_old = Lsyssec
Loop

End

Timerinterrupt:
  Incr _sec
  If _sec = 60 Then
     _sec = 0
     Incr _min
  End If
  If _min = 60 Then
     _min = 0
     Incr _hour
  End If
  If _hour = 24 Then _hour = 0
Return
```

Listing 1 Hello World (hwgmmam32.bas)