MXE11: Unix auf dem Mikrocontroller

V1.73 (c) 2017-2018 Jörg Wolfram

1 Start und Stop

Insgesamt enthält die Imagedatei 3 Diskimages. Alle Images passen in 8 Megabytes und damit auf den AT45DB642 Dataflash. Das erste Disk-Image (/dev/rk0) wird immer nach / gemounted, von den anderen 2 Images kann immer nur eines gemounted werden, vorzugsweise nach /mnt. Das „mount" Binary befindet sich im /etc Ordner, folglich muss der Aufruf dann heißen:

```
/etc/mount /dev/rk1 /mnt
```

und zum unmounten:

```
/etc/umount /dev/rk1
```

Auf Controller-Ebene gibt es einen Cache, der insbesondere häufige Schreibaktionen verhindern soll. Der Write-Cache ist teilweise relativ klein und puffert bei 64K Controller-RAM gerade einmal 4 Sektoren. Das reicht aber aus, um die periodischen Schreibzugriffe im Ruhezustand vom Speichermedium fernzuhalten.

Nach dem Start erscheint kurz eine Meldung, während der man ndurch 2-maliges Drücken der Leertaste in das Setup-Menü gelangt. Momentan lässt sich hier nur die (evtl. vorhandene) RTC stellen. Später können hier noch andere Dinge wie z.B. Image-Transfer über die serielle Schnittstelle dzukommen. Danach wird die aktuelle Version angezeigt und darunter ein Prompt (@). Hier muss jetzt der zu startende Kernel angegeben werden. Beim verwendeten Mini-Unix kann zwischen 2 Kerneln gewählt werden. „rkmx" startet den unveränderten ursprünglichen Kernel, „mx" hingegen einen leicht angepassten. Genaugenommen ändert sich nur die Art, wie Datum und Uhrzeit beim Start bestimmt werden. Während „mx" die aktuelle Zeit der RTC nimmt, liest der „rkmx" Kernel die letzte Änderungszeit des Superblocks auf Disk1. Außerdem kann der modifizierte Kernel mit mehreren Terminals umgehen.

Während des Starts leuchtet die Drive-LED ständig, um danach auszugehen. Falls das Medium nicht gefunden wurde, beginnt die Drive-LED zu blinken und im Terminal wird **MXE11: NO DRIVE** angezeigt. In diesem Fall muss kontrolliert werden, ob das Flashmedium richtig angeschlossen bzw. die SD-Karte kontaktiert ist. Dabei wird nur die physikalische Anwesenheit des Mediums überprüft, nicht der Inhalt.
Danach wird die Sys-LED aktiviert und versucht, vom Medium zu booten. dabei wird der erste Sektor in das RAM eingelesen und an den Anfang des gelesenen Speicherbereiche gesprungen. Ist kein gültiger Bootsektor vorhanden, hält in den meisten Fällen der Emulator mit einem **User Halt** und die Status-LED beginnt zu blinken.

Nach erfolgreichem Start folgt der Login-Prompt. Einloggen kann man sich als **root** oder als User **mxe**, in beiden Fällen ist kein Passwort gesetzt.

Zum Beenden muss die PDP11 heruntergefahren werden. Zu Herunterfahren dient das Kommando bzw. gleichnamige Programm **halt**, der zuerst ein sync() ausführt und danach den Write-Cache im Controller auf das Speichermedium zurückschreibt. Abschließend verlischt die Status-LED und die Stromversorgung kann abgeschaltet werden.

Achtung: Entfernen Sie **NIE** die Stromquelle oder resetten den Controller, solange das System nicht heruntergefahren ist. Geschieht dies trotzdem, ist mit Sicherheit mit Datenverlusten oder inkonsistenten Dateisystemen zu rechnen.

```
#halt
Cleanup cache ......
MXE11: SYSTEM HALTED
```

2 Geänderte und Zusätzliche Programme

2.1 Die Shell (sh)

Die Shell (**sh**) enthält momentan 2 Besonderheiten:

1. Ausser dem unter Unix V6 üblichen **chdir** lässt sich auch **cd** verwenden
2. Mit dem Kommando **prompt** lässt sich (experimentell) der aktuelle Pfad im Prompt einblenden

2.2 System anhalten (halt)

Dieses Programm stammt von mir und läuft nur auf dem MXE11. Es dient zum Herunterfahren des Systems. Zuerst wird ein sync() ausgeführt, dann der Magic-Wert 1234d in die Speicherzelle 0xFABC geschrieben. Dies veranlasst den Emulator, den Cache zu leeren und das System anzuhalten. Nach erfolgtem Halt verlöscht die Status-LED.

2.3 MIPS Benchmark (mips)

Dieser Benchmark führt sehr oft hintereinander den Befehl **INC R4** aus. Als Parameter wird angegeben, wieviele Millionen mal der Befehl ausgeführt werden soll. Gibt man hier z.B. 100 an, lässt sich leicht aus der verstrichenen Zeit der MIPS-Wert bestimmen. Für genauere Resultate sollte der Befehl mittels **time** gestartet werden.

2.4 Türme von Hanoi Benchmark (hanoi)

Dieser Benchmark sortiert einen 10-er Stapel Scheiben um. Der Parameter gibt die Anzahl der Sortierungen an, ein guter wert ist 1000. Für genaue Resultate sollte der Befehl mittels **time** gestartet werden, die Anzahl der Loops pro Sekunde ist dann
1000 / gemessene Zeit.

3 Spezielle Adressen im I/O Raum

Zusätzlich zur „normalen" Peripherie einer PDP11 gibt es beim MXE11 zusätzliche Funktionen, die über I/O Adressen angesprochen werden. Dabei ist die Zugriffsbreite (SIZE) zu beachten, die entweder BYTE oder WORD ist. Ein Zugriff auf eine WORD-Adresse über Byte-Zugriffe ist dabei nicht möglich, Gleiches gilt für den umgekehrten Fall. Wenn beide Zugriffsarten erlaubt sind, gibt es in den Tabellen auch zwei Einträge. Hinter der Zugriffsbreite steht noch die zugehörige Aktion (RD=Lesen, WR=Schreiben).

3.1 Shutdown

Um den Rechner herunterfahren zu können, ist es notwendig, auch den im Mikrocontroller befindlichen Cache zu leeren. Dies wird ausgelöst, wenn das „magische" Word 0x1234 an die Adresse 0xFABC geschrieben wird.

ADDR	SIZE	Funktion
0175274 (0xFABC)	WORD (WR)	Schreiben von 0x1234 löst Shutdown aus

Vorher sollte aber mittels **sync()** der Cache im Unix-system geleert werden. Dies ist z.B. beim **halt** Kommando so realisiert:

```
int main()
{
    int *ptr;
    sync();
    ptr = 0175274;
    *ptr = 011064;
    return 0;
}
```

3.2 RTC

Sofern eine RTC angeschlossen ist, kann der aktuelle Unix Timestamp ausgelesen werden. Ist keine RTC vorhanden, wird bei beiden Adressen der Wert 0xFFFF zurückgegeben.

ADDR	SIZE	Funktion
0175260 (0xFAB0)	WORD (RD)	liest RTC aus und gibt HIGH-Word zurück
0175262 (0xFAB2)	WORD (RD)	gibt HIGH-Word des RTC-Wertes zurück

3.3 Zufallszahlen (ab v1.73)

Ab Version 1.73 gibt es einen (Pseudo-) Zufallszahlengenerator. Dabei ist die Zahlenfolge nach Systemstart nicht reproduzierbar, da auch 7500 mal pro Sekunde bei jedem Tick-Interrupt ein neuer Zufallswert berechnet (und wieder verworfen) wird.

ADDR	SIZE	Funktion
0175420 (0xFB10)	BYTE (RD)	liest ein neues Zufallsbyte
0175420 (0xFB10)	WORD (RD)	liest ein neues Zufallswort

Zufallwerte können auch über das **/dev/random** Device abgefragt werden. Zusätzlich gibt es noch ein **/dev/orandom** Device, hier werden jedoch kontinuierlich Oktalziffern (0..7) als ASCII-Werte ausgegeben.

```
dd if=/dev/orandom of=/dev/tty8 bs=1 count=40
```

gibt 40 Zufallsziffern nacheinander auf der Konsole aus.

3.4 Tick-Counter (ab v1.73)

Ab Version 1.73 gibt es auch einen Tick-Counter. Dieser ist 16 Bit breit und zählt bei jedem Tick-Interrupt um 1 weiter. Der Tick-Interrupt läuft mit 12KHz.

ADDR	SIZE	Funktion
0175422 (0xFB12)	WORD (RD)	liest den aktuellen Zählerstand
0175422 (0xFB12)	WORD (WR)	setzt den Zähler auf den angegebenen Wert

Mittels Tick können z.B. Aktionen in einem feineren Zeitraster ausgeführt werden. Das funktioniert aber nur bei den Mikrocontroller-Varianten.

```
int *ptr;

ptr = 0175422;
*ptr = 0;
while(*ptr < 12);
```

3.5 Digital Input (ab v1.73)

Die Controller-Versionen von MXE11 sellen 4 digitale Input-Pins zur Verfügung, die genaue Belegung ist im Dokument über die Hardware beschrieben.

ADDR	SIZE	Funktion
0175454 (0xFB2C)	BYTE (RD)	liest digital I/O ein (Bit 0–3)

Der Status der 4 Eingänge (DIN0-DIN3) liegt in den Bits 0-3, die Bits 4-7 sind immer 0. Je nach Controller kann es sein, dass die Eingänge nicht exakt gleichzeitig abgefragt werden, gegenüber der maximalen Auslesefrequenz sollte das aber in den meisten Fällen eine untergeordnete Rolle spielen.

Die digitalen Inputs können auch über das **/dev/gpio** Device abgefragt werden, hier wird allerdings ein Hex-Char zurückgegeben.

```
dd if=/dev/gpio of=/dev/tty8 bs=1 count=20
```

gibt 20 Werte nacheinander auf der Konsole aus. Bei den SDL-Varianten ist der Wert immer 1 (willkürlich festgelegt).

3.6 Digital Output (ab v1.73)

Die Controller-Versionen von MXE11 sellen 4 digitale Output-Pins zur Verfügung, die genaue Belegung ist im Dokument über die Hardware beschrieben.

ADDR	SIZE	Funktion
0175474 (0xFB3C)	BYTE (WR)	setzt die digitalen Ausgänge (Bit 0–3)
0175474 (0xFB3C)	BYTE (RD)	liest aktuellen Zustand der digitalen Ausgänge (Bit 0–3)

Der zu setzende Zustan für die 4 Ausgänge (DOUT0-DOUT3) liegt in den Bits 0-3, die Bits 4-7 werden ignoriert. Je nach Controller kann es sein, dass die Eingänge nicht exakt gleichzeitig gesetzt werden, gegenüber der maximalen Setzfrequenz sollte das aber in den meisten Fällen eine untergeordnete Rolle spielen.

Die digitalen Outputs können auch über das **/dev/gpio** Device gesetzt werden, hier wird allerdings ein Hex-Char erwartet.

```
echo 5 > /dev/gpio
```

setzt die Ausgänge DOUT0 und DOUT2, die Ausgänge DOUT1 und DOUT3 werden zurückgesetzt.

3.7 Analog Input (ab v1.73)

Die Controller-Versionen von MXE11 sellen 4 analoge Input-Pins (AIN0-AIN3) mit 10 Bit Auflösung zur Verfügung, die genaue Belegung ist im Dokument über die Hardware beschrieben.

ADDR	SIZE	Funktion
0175440 (0xFB20)	WORD (RD)	Wert von AIN0 (10 Bit)
0175442 (0xFB22)	WORD (RD)	Wert von AIN1 (10 Bit)
0175444 (0xFB24)	WORD (RD)	Wert von AIN2 (10 Bit)
0175446 (0xFB26)	WORD (RD)	Wert von AIN3 (10 Bit)

Die digitalen Inputs können auch über die **/dev/adcx** Devices abgefragt werden, hier wird allerdings ein mit LF terminierter String zurückgegeben.

```
dd if=/dev/adc0 of=/dev/tty8 bs=1 count=20
```

gibt 4 Werte untereinander auf der Konsole aus. Bei den SDL-Varianten ist der Wert immer 0111 für AIN0 bis 0444 für AIN3.

3.8 PWM/Analog Output (ab v1.73)

Die Controller-Versionen von MXE11 sellen 4 PWM Output-Pins (AOUT0-AOUT3) zur Verfügung, die genaue Belegung ist im Dokument über die Hardware beschrieben.

ADDR	SIZE	Funktion
0175460 (0xFB30)	WORD (WR)	Wert von AOUT0 (10 Bit)
0175462 (0xFB32)	WORD (WR)	Wert von AOUT1 (10 Bit)
0175464 (0xFB34)	WORD (WR)	Wert von AOUT2 (10 Bit)
0175466 (0xFB36)	WORD (WR)	Wert von AOUT3 (10 Bit)

Die analogen Outputs können auch über die **/dev/pwmx** Devices gesetzt werden, hier wird ein numerischer String erwartet, bei Leerzeichen oder LF wird der Wert ausgegeben.

```
echo 512 > /dev/pwm1
```

setzt den Ausgang AOUT1 auf 50% PWM.