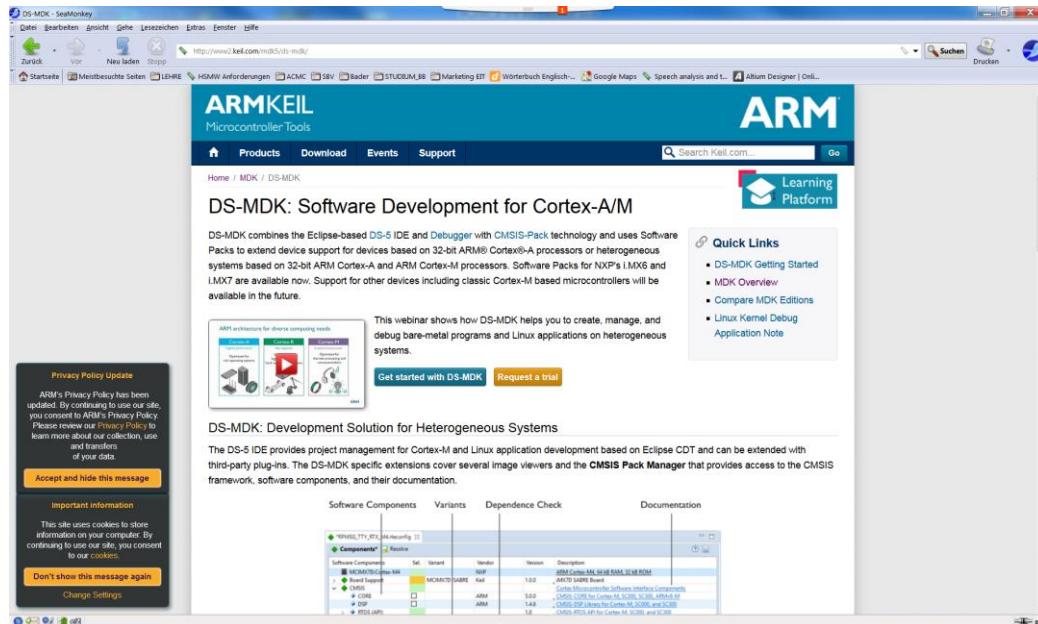


# 1. Verwendete Programmierumgebung

WEB: [www2.keil.com/mdk5/ds-mdk/](http://www2.keil.com/mdk5/ds-mdk/)



## Installation auf eigenem Rechner

Installationsdatei unter R:\CB\Bader\PR HWPR\KEIL MDK5

- MDK521a.EXE (Installationsdatei)
- MDKCM521.EXE (Backward-Support, erst nach Installation von KEIL MDK5.21 ausführen)

## Programmierung/ Test

The screenshot shows the IAR Embedded Workbench IDE interface. The main window displays a C code file named "main.c" for a project titled "I\_0\_LUGRModul HWPR\projekte\I\_0\_blinky\_1\_einfach\_mit\_display\blinky\_einfach\proj\jvision". The code implements a simple LED blink sequence using the LPC\_GPIO1 peripheral. The code includes comments explaining the setup of pins P1.18 and P1.19 as outputs, and the toggling of pins P1.18, P1.19, P1.20, and P1.21 in a loop. The IAR interface also shows a "Build Output" window at the bottom with log messages related to the build and flash process.

```

53 // 1. Richtung Port auf Ausgang schalten
54 // LED0 ist über die define Anweisung die 18 zugeordnet und die 1 wird 18 mal geschoben um
55 // Pin P1.18 wo erste led angeschlossen auf Ausgang zu programmieren
56 // dasselbe mit den anderen Leds
57
58 LPC_GPIO1->FIODIR |= (1<<LED0); // LED0 bedeutet 18 über define Anweisung und die 1 wird 18 mal geschoben um den Port auf Ausgang zu setzen
59 LPC_GPIO1->FIODIR |= (1<<LED1);
60 LPC_GPIO1->FIODIR |= (1<<LED2);
61 LPC_GPIO1->FIODIR |= (1<<LED3);
62
63 // clear pins ( turning of the leds
64 // hier wird der Ausgang auf low geschaltet Led aus
65
66 LPC_GPIO1->FIOCLR |= (1<<LED0);
67 LPC_GPIO1->FIOCLR |= (1<<LED1);
68 LPC_GPIO1->FIOCLR |= (1<<LED2);
69 LPC_GPIO1->FIOCLR |= (1<<LED3);
70
71
72 while(1)
73 {
74
75     if (izustand==0)
76     {
77         LPC_GPIO1->FIOSET |= (1<<LED2); // Led s ein schalten 2, 3
78         LPC_GPIO1->FIOSET |= (1<<LED3);
79         izustand=1;
80         Warte(1000);
81     }
82     else
83     {
84         LPC_GPIO1->FIOCLR |= (1<<LED2); // Led ausschalten
85         LPC_GPIO1->FIOCLR |= (1<<LED3);
86         izustand=0;
87         Warte(1000);
88     }
89
90 } // ende while
91 } // ende main
92

```

## Debugging

The screenshot shows the IAR Embedded Workbench IDE interface. The assembly window displays the code for the main function, with several instructions highlighted in yellow. The C source code window shows the corresponding C code. The Registers window shows the CPU register values, and the GPIO window shows the port pin values. The Watch window displays the value of the variable 'izustand'.

```

IAR Embedded Workbench - Project: HWPP\projecte\3_blinky_1_einfach_mit_display\blinky_einfach\uproj - v1.000

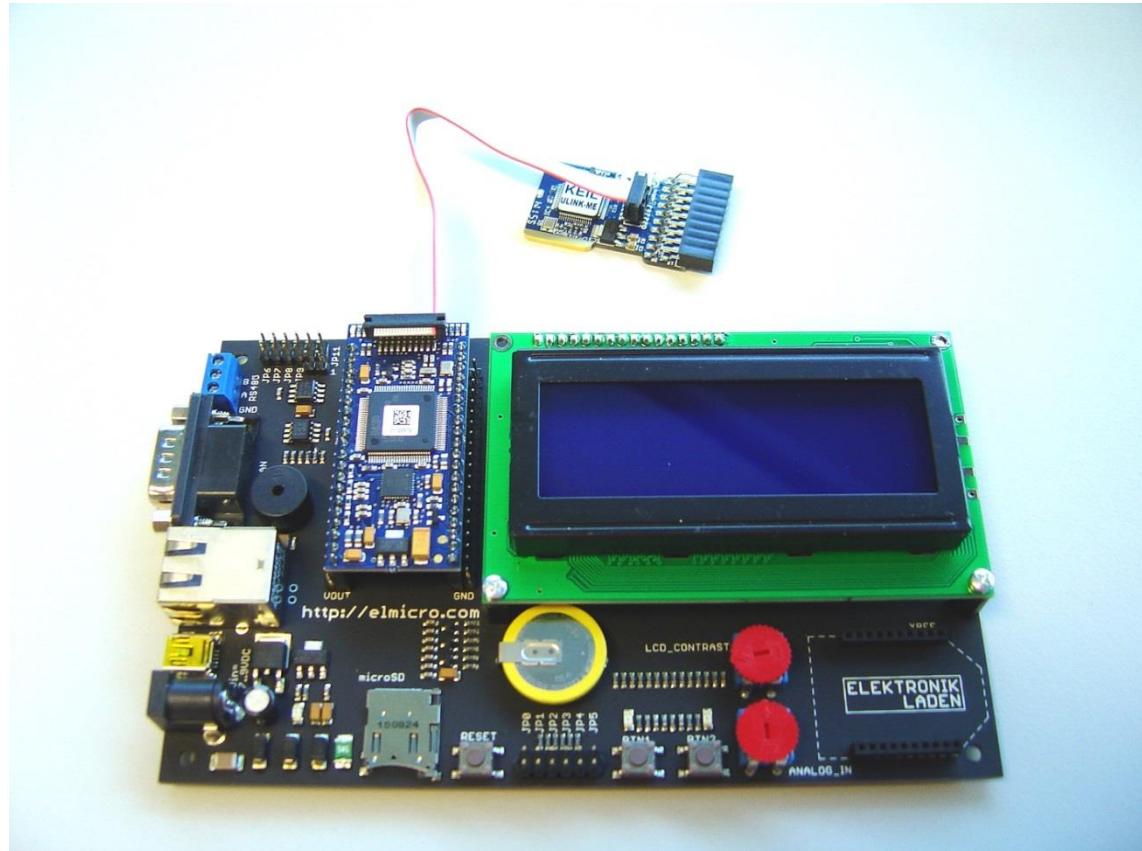
Registers
Register Value
Core
R0 0x00000000
R1 0x00002710
R2 0x000000E8
R3 0x00002710
R4 0x00000000
R5 0x2007C004
R6 0x00000001
R7 0x000000E8
R8 0x000000E8
R9 0x100001DC
R10 0x00000000
R11 0x00000000
R12 0x0007C048
R13 (SP) 0x10000000
R14 (LR) 0x00000045
R15 (PC) 0x00000056

main.c
73 {
74     if (izustand==0)
75     {
76         if (LPC_GPIO1->FIOSSET |= (1<<LED2)); // Led s ein schalten 2, 3
77         izustand=1;
78     }
79     else
80     {
81         if (LPC_GPIO1->FIOSCLR |= (1<<LED2)); // Led ausschalten
82         izustand=0;
83     }
84 }
85 } // ende while
86 } // ende main
87
88
89
90
91
92
93
94 int warte(int izahl)
95 {

```

## 2. Verwendete Hardware

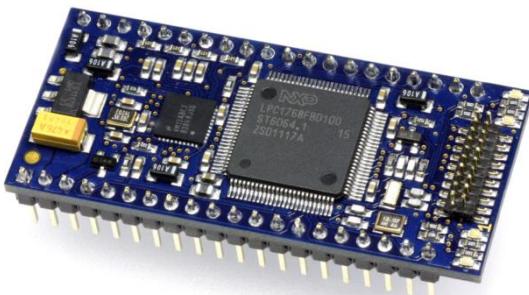
### TestBed for mbed



## On-Bord-Funktionen

- 2 Eingabetaster
- 2 LEDs
- 3 zeiliges alphanumerisches LCD-Display
- Piezo-Lautsprecher
- 1 Potentiometer für Analogspannung
- micro-SD Karten Slot
- 10/100MBit Ethernet Anschluss
- RS485-Schnittstelle
- CAN – Schnittstelle
- UART-Anschluss
- mini-USB Buchse
- Backup-Batterie für RTC (Real Time Clock)
- Reset-Taster
- XBee-Sockel
- Steckplatz für Arduino Shields
- 2,54mm Prototyping-Fläche

## MC-Modul Chip1768



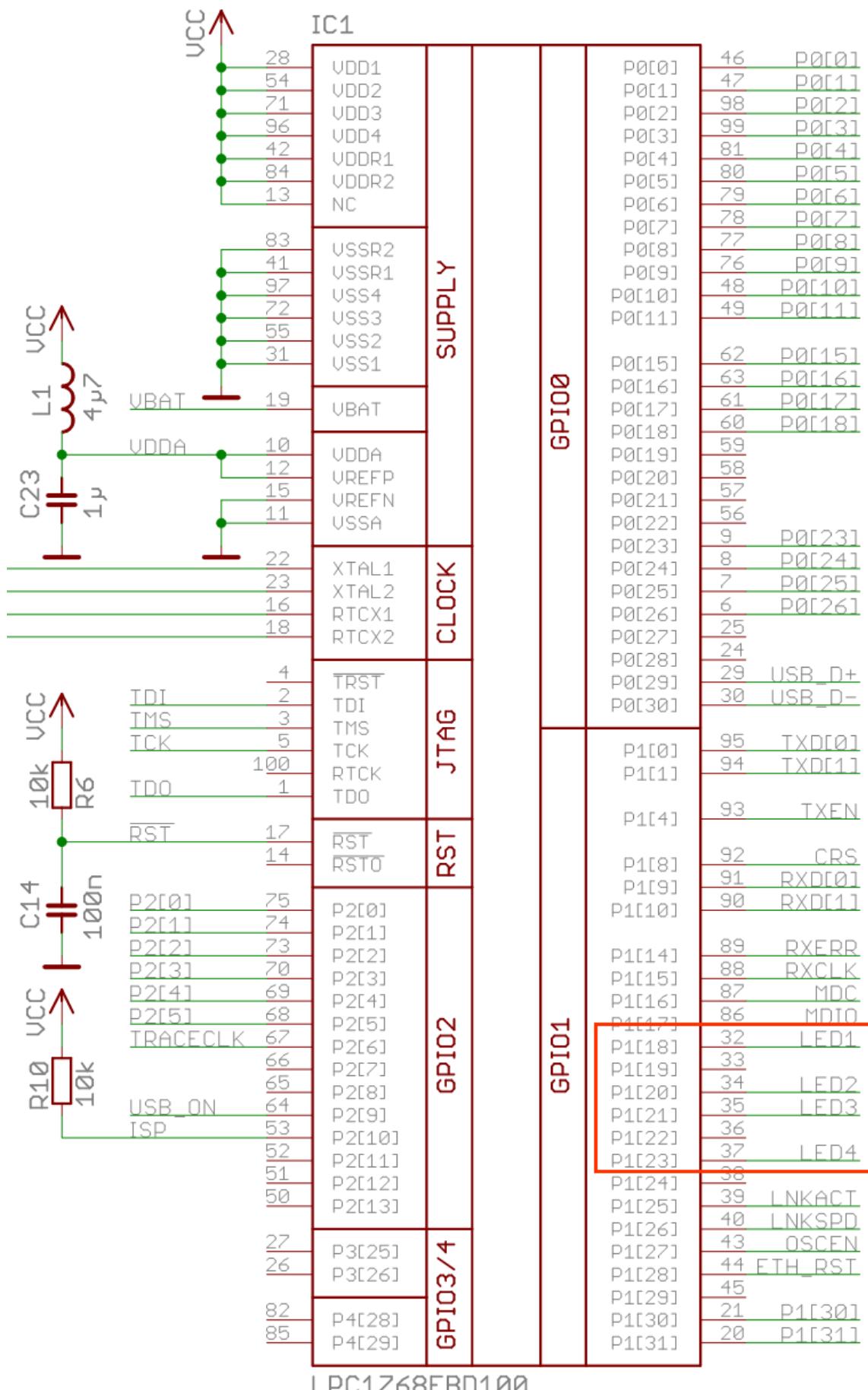
Modul „chip1768“

## On-Module Funktionalität

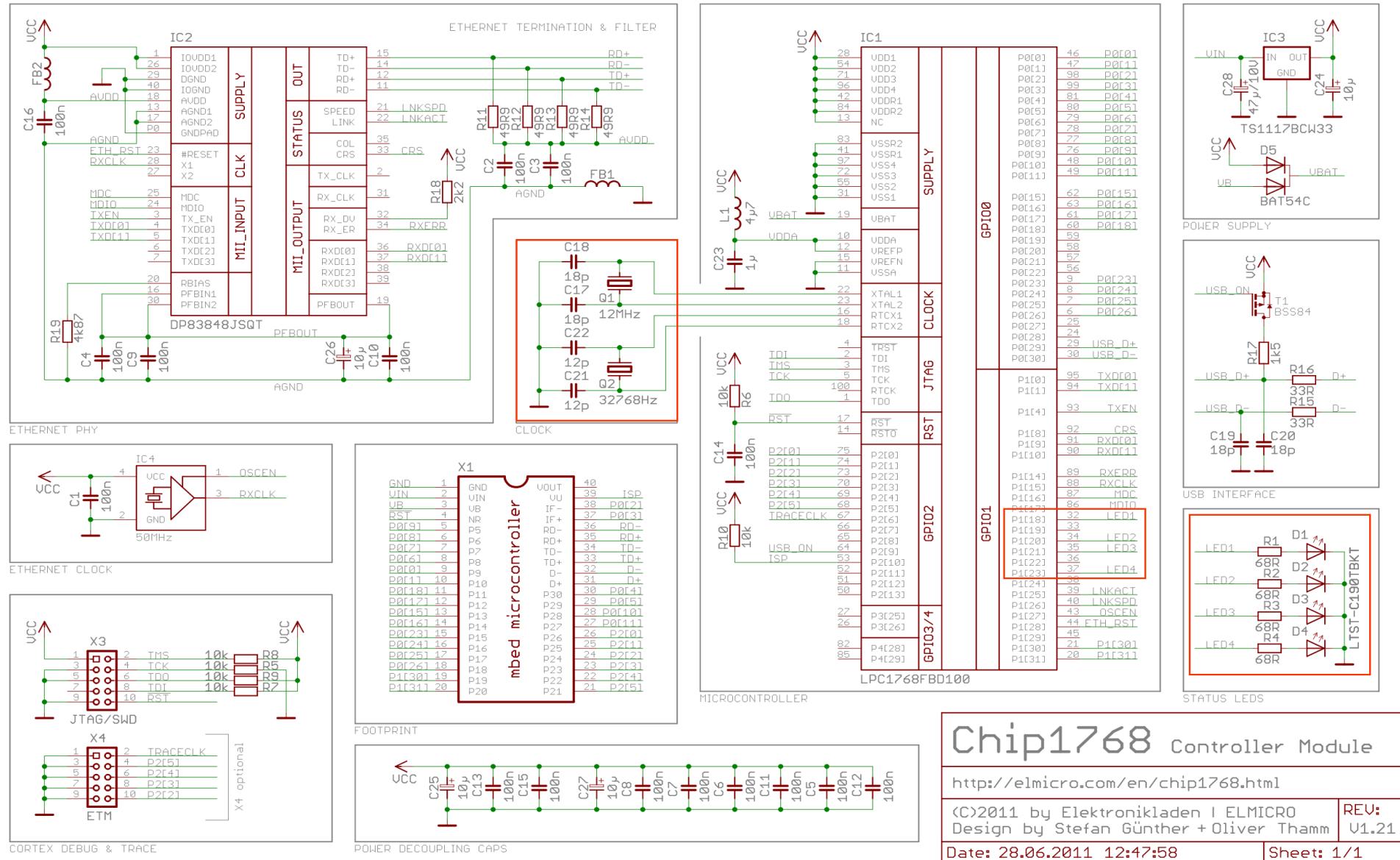
- Mikrocontroller NXP LPC1768 MCU
  - 512 kB FLASH
  - 2x32 kB RAM
- 12MHz Quarz für CPU (ergibt  $\text{CPU}_{\max} = 100 \text{ MHz}$ )
- 32,786kHz Quarz für RTC
- 4 Signal-LEDs
- National Semiconductor DP83848J Ethernet Transceiver
- Cortex-Debug-Schnittstelle (Unterstützt JTAG, SWD, 4-Bit Trace),  
(Stiftleiste 2x10 RM 1,27mm)
- Eingangsspannungsbereich 4,5V .. 9V; Betriebsspannung 3,3V
- max. Stromaufnahme ca. 200mA  
(bei Versorgungsspannung von 5V und Nutzung der Ethernet-Schnittstelle)
- 40-polige DIP-Bauform RM 2,54mm, Sockelbelegung für mBed/TestBed
- Gesamtabmessungen: 56,50mm x 25,90mm

## Mikrocontroller NXP LPC1768 MCU

- ARM Cortex-M3 (ARMv7) 32-Bit CPU
- bis 100MHz Haupttaktfrequenz
- 512kB Programmspeicher (Flash), 2x32kB RAM
- ARM Cortex Trace Unterstützung (Trace Port Unit)
- 10/100MBit Ethernet, RMII Interface, DMA-Controller
- 12-Bit Analog-Digital-Wandler mit 8 Kanälen
- 10-Bit Digital-Analog-Wandler (1 Kanal)
- 4x 32-Bit Timer
- 6 PWM-Kanäle, 1x Motor Control PWM, 8 DMA-Kanäle
- USB 2.0 Interface mit integriertem Transceiver
- CAN2.0B mit 2 Kanälen
- 4x UART, 2x SSP, 1x SPI, 3x I2C, 1x I2S
- Interface für Quadratur-Encoder
- Low Power RTC, Unique ID
- interner 4MHz RC-Oszillator

**LPC1768, Pinbeschaltung (Auszug Schaltplan, alle Pins dargestellt)**

## LPC Modul, Schaltplan



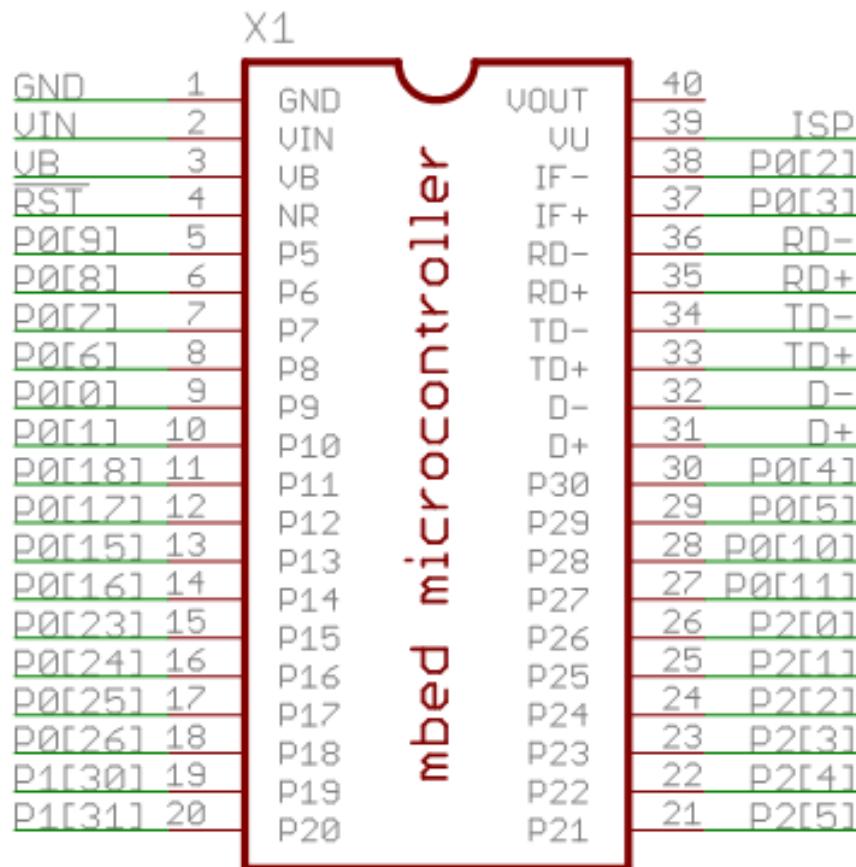
Chip1768 Controller Module

<http://elmicro.com/en/chip1768.html>(C)2011 by Elektronikladen I ELMICRO  
Design by Stefan Günther + Oliver ThammREV:  
V1.21

Date: 28.06.2011 12:47:58

Sheet: 1/1

### Belegung Stecksockel TestBed/mBed



Signal	Typ	Beschreibung
GND	-	Schaltungsmasse
VIN	Eingang	4,5 .. 9V
VB	Eingang	3V Batteriespannung für RTC
!RST	Eingang, low-aktiv	Haupt-Resetsignal für LPC1768
P5..P30	Ein-/Ausgang	Multifunktions-Ports, siehe Tabelle 2.2
D+	Ein-/Ausgang	USB-Datenleitung
D-	Ein-/Ausgang	USB-Datenleitung
TD+	Ausgang	Ethernet Sendeleitung, Positivsignal
TD-	Ausgang	Ethernet Sendeleitung, Negativsignal
RD+	Eingang	Ethernet Empfangsleitung, Positivsignal
RD-	Eingang	Ethernet Empfangsleitung, Negativsignal
IF+	Eingang	Empfangsleitung für Bootloader
IF-	Ausgang	Sendeleitung für Bootloader
ISP	Eingang	Steuereingang für den Bootloader-Modus

### ACHTUNG!

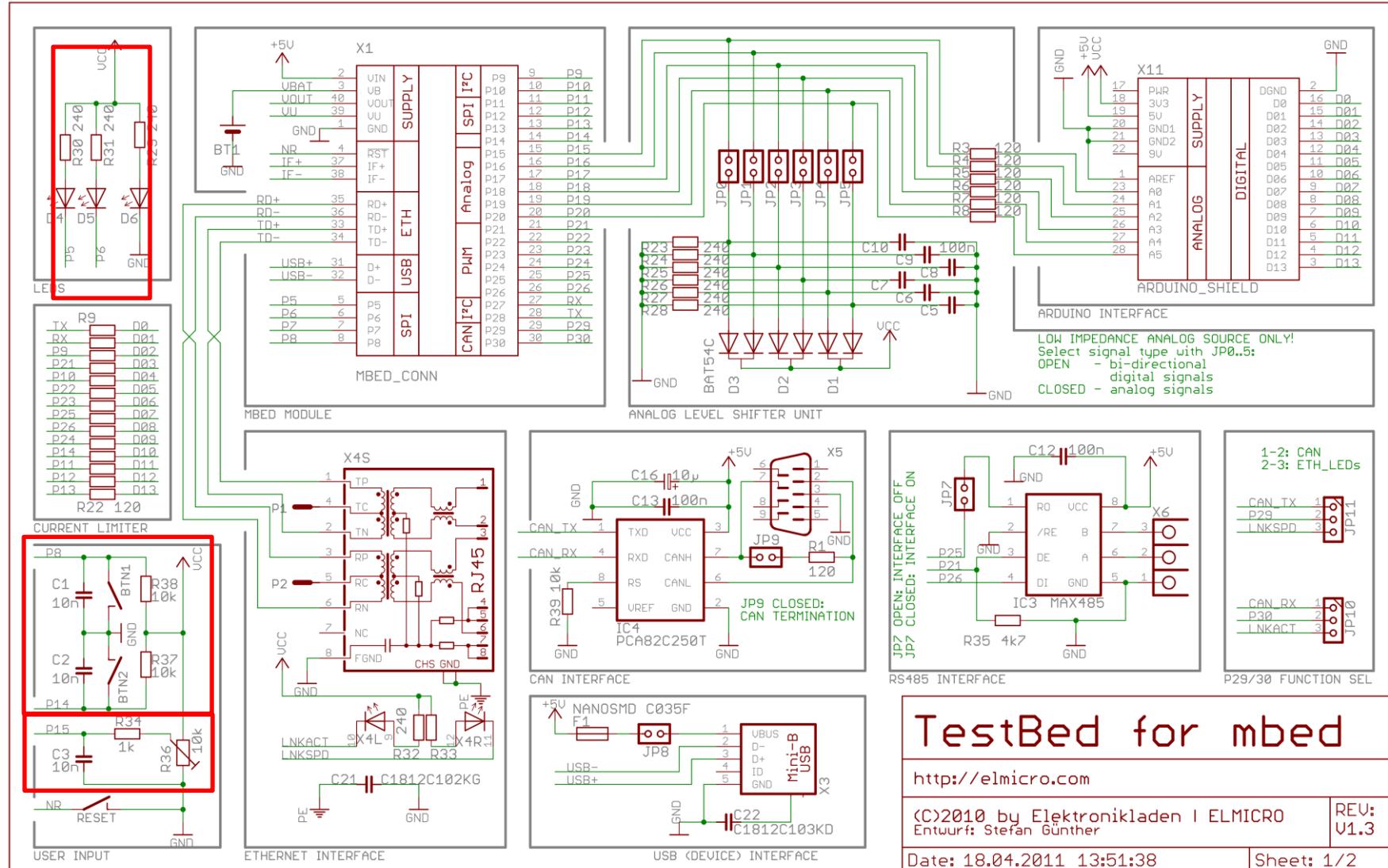
Nicht alle Pins des LPC1786 sind über den Stecksockel herausgeführt und damit für das Testbord (TestBed) nutzbar.

Kontakte des Stecksockels (Pin) und daran angeschlossenes Portpins des LPC1768 (1. Funktion), sowie weitere, zur Nutzung stehende Funktionen des Controllers.

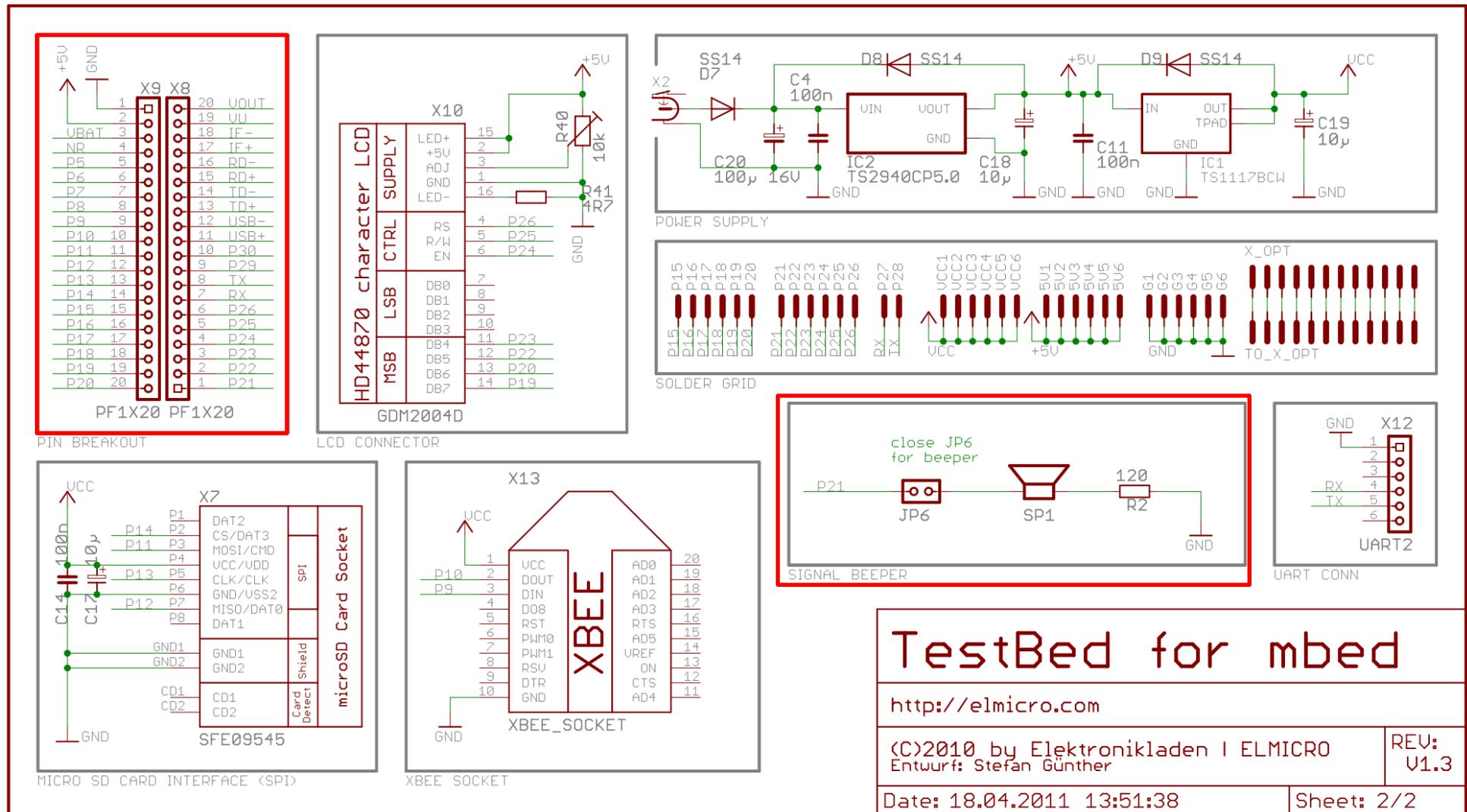
<u>TestBed</u>	<u>Pin</u>	<u>1. Funktion</u>	<u>2. Funktion</u>	<u>3. Funktion</u>	<u>4. Funktion</u>	<u>Pin am LPC1768</u>
LED4	<i>P5</i>	P0[9]	I <sup>2</sup> S (TX_SDA)	SSP1 (MOSI)	MAT2[3]	76
LED5	<i>P6</i>	P0[8]	I <sup>2</sup> S (TX_WS)	SSP1 (MISO)	MAT2[2]	77
	<i>P7</i>	P0[7]	I <sup>2</sup> S (TX_CLK)	SSP1 (SCK)	MAT2[1]	78
BTN1	<i>P8</i>	P0[6]	I <sup>2</sup> S (RX_SDA)	SSP1 (SSEL)	MAT2[0]	79
	<i>P9</i>	P0[0]	CAN1 (RD1)	UART3 (TX)	I <sup>2</sup> C1 (SDA)	46
	<i>P10</i>	P0[1]	CAN1 (TD1)	UART3(RX)	I <sup>2</sup> C1 (SCL)	47
	<i>P11</i>	P0[18]	UART1 (DCD)	SSP0 (MOSI)	SPI (MOSI)	60
	<i>P12</i>	P0[17]	UART1 (CTS)	SSP0 (MISO)	SPI (MISO)	61
	<i>P13</i>	P0[15]	UART1 (TXD)	SSP0 (SCK)	SPI (SCK)	62
BTN2	<i>P14</i>	P0[16]	UART1 (RXD)	SSP0 (SSEL)	SPI (SSEL)	63
ANALOG_IN	<i>P15</i>	P0[23]	A/D0[0]	I <sup>2</sup> S (RX_CLK)	CAP3[0]	9
	<i>P16</i>	P0[24]	A/D0[1]	I <sup>2</sup> S (RX_WS)	CAP3[1]	8
	<i>P17</i>	P0[25]	A/D0[2]	I <sup>2</sup> S (RX_SDA)	UART3 (TXD)	7
	<i>P18</i>	P0[26]	A/D0[3]	DAC out	UART3 (RXD)	6
	<i>P19</i>	P1[30]	V <sub>BUS</sub>	A/D0[4]		21
	<i>P20</i>	P1[31]	SSP1 (SCK)	A/D0[5]		20
Speaker+JP6	<i>P21</i>	P2[5]	PWM1[6]	UART1 (DTR)	TRACEDATA[0]	68
	<i>P22</i>	P2[4]	PWM1[5]	UART1 (DSR)	TRACEDATA[1]	69
	<i>P23</i>	P2[3]	PWM1[4]	UART1 (DCD)	TRACEDATA[2]	70
	<i>P24</i>	P2[2]	PWM1[3]	UART1 (CTS)	TRACEDATA[3]	73
	<i>P25</i>	P2[1]	PWM1[2]	UART1 (RXD)		74
	<i>P26</i>	P2[0]	PWM1[1]	UART1 (TXD)		75
	<i>P27</i>	P0[11]	UART2 (RXD)	I <sup>2</sup> C2 (SCL)	MAT3[1]	49
	<i>P28</i>	P0[10]	UART2 (TXD)	I <sup>2</sup> C2 (SDA)	MAT3[0]	48
	<i>P29</i>	P0[5]	I <sup>2</sup> S (RX_WS)	CAN2 (TD)	CAP2[1]	80
	<i>P30</i>	P0[4]	I <sup>2</sup> S (RX_CLK)	CAN2 (RD)	CAP2[0]	81
	<i>D+</i>	P0[29]	USB (D+)			29
	<i>D-</i>	P0[30]	USB (D-)			30
	<i>RX+</i> <i>RX-</i> <i>TX+</i> <i>TX-</i>	(Ethernet)				-
	<i>IF+</i>	P0[2]	UART0 (TX)	A/D0[7]		98
	<i>IF-</i>	P0[3]	UART0 (RX)	A/D0[6]		99
	<i>ISP</i>	P2[10]	!EINT0	NMI		41

## Schaltung TestBed for mbed

Teil1



## Teil2



### 3. On Chip Komponenten LPC1768

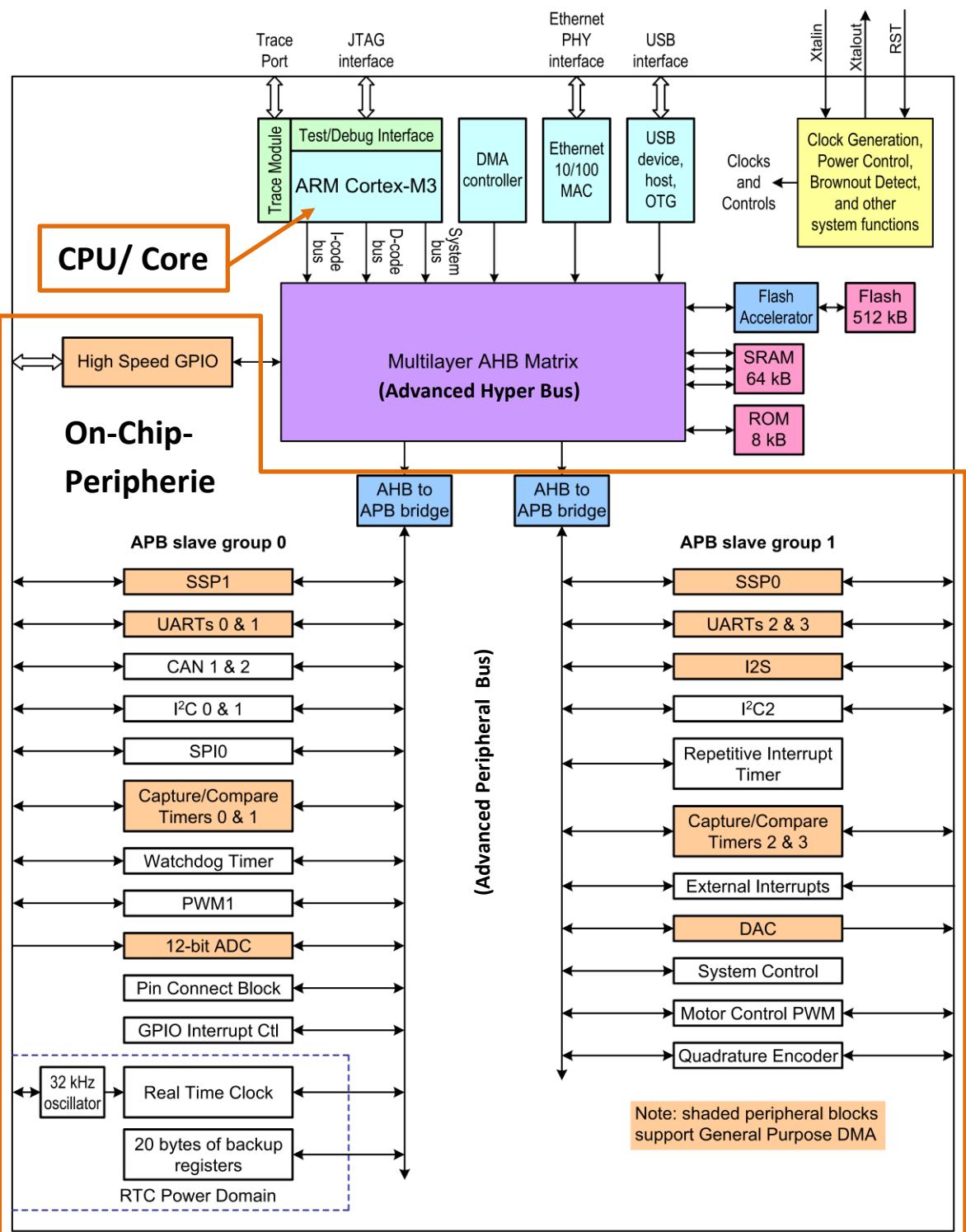
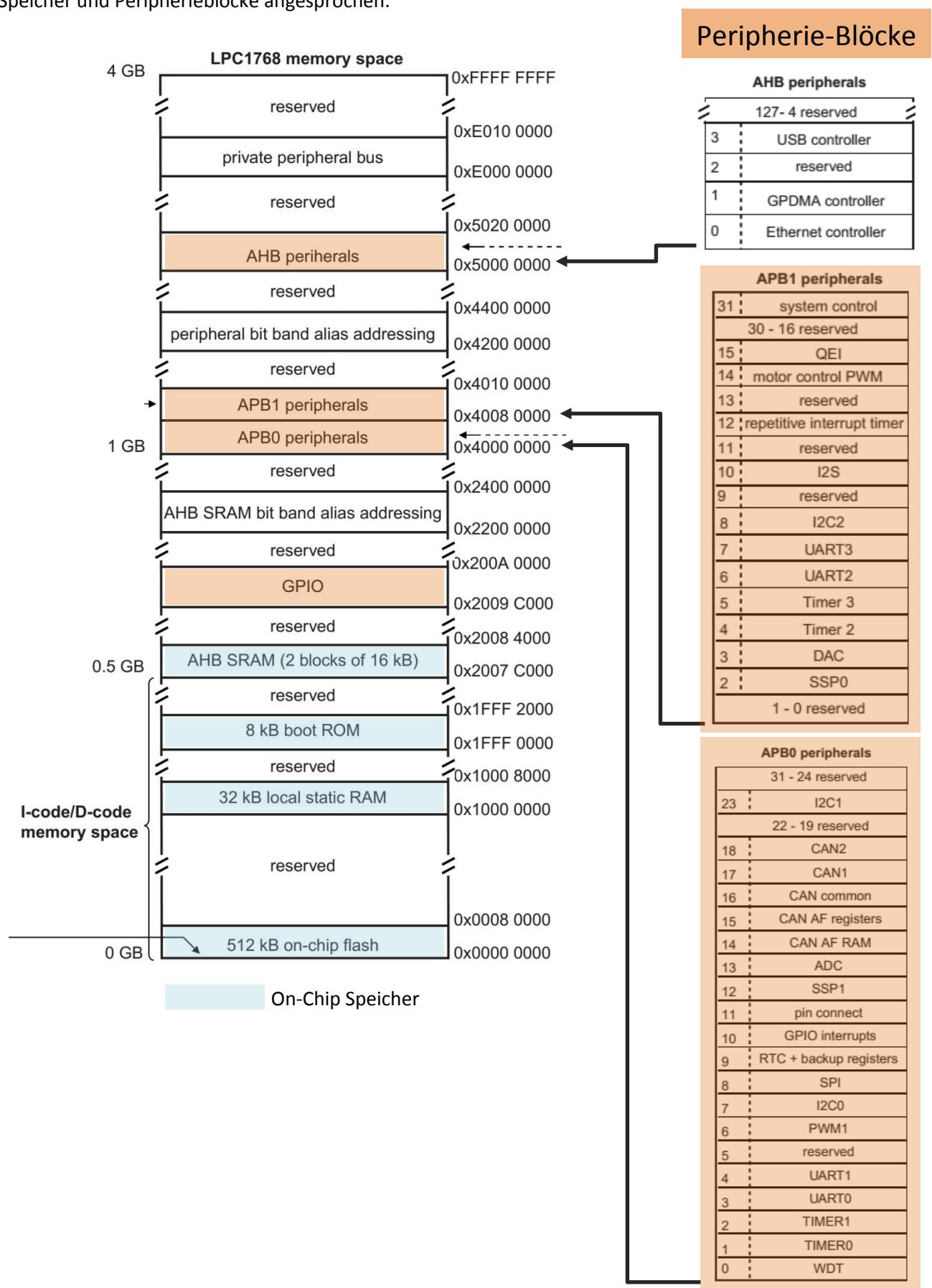


Fig 1. LPC1768 simplified block diagram

## 4. System Memory Map

Der LPC1768 besitzt einen linearen Adressraum von 0000 0000 bis FFFF FFFF. Über diesen werden alle Speicher und Peripherieblöcke angesprochen.



## 5. Peripheral map

Die On-Chip-Peripherie-Hardware wird über Konfigurationsregister eingestellt. Diese sind den Peripherie-Blöcken zugeordnet. Die Grafik zeigt die Peripherie-Blöcke mit den zugehörigen Anfangsadressen. Alle darin befindlichen Peripherie-Register sind 32Bit breit. Compilerhersteller oder Chiphersteller bieten oft Funktionsbibliotheken an, welche die Adressen der Peripherie-Register zuweisen (z.B. LPC17xx.h, KEIL)

High Speed GPIO		Auszug LPC17xx.h :
GPIO		/* GPIOs */
		#define LPC_GPIO0_BASE (LPC_GPIO_BASE + 0x00000)
		#define LPC_GPIO1_BASE (LPC_GPIO_BASE + 0x00020)
		#define LPC_GPIO2_BASE (LPC_GPIO_BASE + 0x00040)
APB0 peripherals		/* APB0 peripherals */
31 - 24 reserved		#define LPC_WDT_BASE (LPC_APB0_BASE + 0x00000)
23	I2C1	#define LPC_TIM0_BASE (LPC_APB0_BASE + 0x04000)
22 - 19 reserved		#define LPC_TIM1_BASE (LPC_APB0_BASE + 0x08000)
18	CAN2	#define LPC_UART0_BASE (LPC_APB0_BASE + 0x0C000)
17	CAN1	#define LPC_UART1_BASE (LPC_APB0_BASE + 0x10000)
16	CAN common	#define LPC_PWM1_BASE (LPC_APB0_BASE + 0x18000)
15	CAN AF registers	#define LPC_I2C0_BASE (LPC_APB0_BASE + 0x1C000)
14	CAN AF RAM	#define LPC_SPI_BASE (LPC_APB0_BASE + 0x20000)
13	ADC	#define LPC_RTC_BASE (LPC_APB0_BASE + 0x24000)
12	SSP1	#define LPC_GPIOINT_BASE (LPC_APB0_BASE + 0x28080)
11	pin connect	#define LPC_PINCON_BASE (LPC_APB0_BASE + 0x2C000)
10	GPIO interrupts	#define LPC_SSP1_BASE (LPC_APB0_BASE + 0x30000)
9	RTC + backup registers	#define LPC_ADC_BASE (LPC_APB0_BASE + 0x34000)
8	SPI	#define LPC_CANAF_RAM_BASE (LPC_APB0_BASE + 0x38000)
7	I2C0	#define LPC_CANAF_BASE (LPC_APB0_BASE + 0x3C000)
6	PWM1	#define LPC_CANCR_BASE (LPC_APB0_BASE + 0x40000)
5	reserved	#define LPC_CAN1_BASE (LPC_APB0_BASE + 0x44000)
4	UART1	#define LPC_CAN2_BASE (LPC_APB0_BASE + 0x48000)
3	UART0	#define LPC_I2C1_BASE (LPC_APB0_BASE + 0x5C000)
2	TIMER1	
1	TIMER0	
0	WDT	

APB1 peripherals		Auszug LPC17xx.h :
0x4010 0000		
0x400F C000	31 : system control	/* APB1 peripherals */
0x400C 0000	30 - 16 reserved	#define LPC_SSP0_BASE (LPC_APB1_BASE + 0x08000)
0x400B C000	15 : QEI	#define LPC_DAC_BASE (LPC_APB1_BASE + 0x0C000)
0x400B 8000	14 : motor control PWM	#define LPC_TIM2_BASE (LPC_APB1_BASE + 0x10000)
0x400B 4000	13 : reserved	#define LPC_TIM3_BASE (LPC_APB1_BASE + 0x14000)
0x400B 0000	12 : repetitive interrupt timer	#define LPC_UART2_BASE (LPC_APB1_BASE + 0x18000)
0x400A C000	11 : reserved	#define LPC_UART3_BASE (LPC_APB1_BASE + 0x1C000)
0x400A 8000	10 : I2S	#define LPC_I2C2_BASE (LPC_APB1_BASE + 0x20000)
0x400A 4000	9 : reserved	#define LPC_I2S_BASE (LPC_APB1_BASE + 0x28000)
0x400A 0000	8 : I2C2	#define LPC_RIT_BASE (LPC_APB1_BASE + 0x30000)
0x4009 C000	7 : UART3	#define LPC_MCPWM_BASE (LPC_APB1_BASE + 0x38000)
0x4009 8000	6 : UART2	#define LPC_QEI_BASE (LPC_APB1_BASE + 0x3C000)
0x4009 4000	5 : Timer 3	#define LPC_SC_BASE (LPC_APB1_BASE + 0x7C000)
0x4009 0000	4 : Timer 2	
0x4008 C000	3 : DAC	
0x4008 8000	2 : SSP0	
0x4008 0000	1 - 0 reserved	

**Bezeichner für Peripherie-Blöcke in LPC17xx.h (Auszug)**

Auszug aus der Peripheral declaration :

LPC_SC	LPC_GPIOINT
LPC_GPIO0	LPC_PINCON
LPC_GPIO1	LPC_SSP0
LPC_GPIO2	LPC_SSP1
LPC_GPIO3	LPC_ADC
LPC_GPIO4	LPC_DAC
LPC_WDT	LPC_CANAF_RAM
LPC_TIM0	LPC_CANAF
LPC_TIM1	LPC_CANCR
LPC_TIM2	LPC_CAN1
LPC_TIM3	LPC_CAN2
LPC_RIT	LPC_MCPWM
LPC_UART0	LPC_QEI
LPC_UART1	LPC_EMAC
LPC_UART2	LPC_GPDMA
LPC_UART3	LPC_GPDMAH0
LPC_PWM1	LPC_GPDMAH1
LPC_I2C0	LPC_GPDMAH2
LPC_I2C1	LPC_GPDMAH3
LPC_I2C2	LPC_GPDMAH4
LPC_I2S	LPC_GPDMAH5
LPC_SPI	LPC_GPDMAH6
LPC_RTC	LPC_GPDMAH7
	LPC_USB

Das Schreiben des zugewiesenen Bezeichners im Quellcode mit einem Zeigeroperator(>), bewirkt in µVision das Öffnen eines Auswahlmenüs mit den Peripherie-Registern des Blocks.

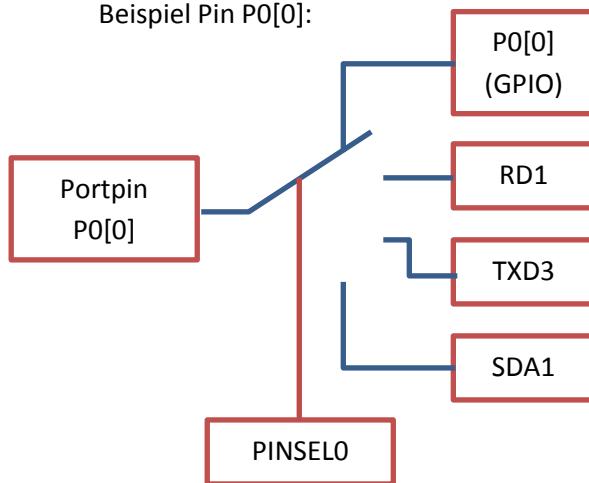
## 6. Port Ein-/Ausgabe

- 3 Ports mit 32 Bit (P0, 1, 2, nicht alle Bits verfügbar)
- 4 Hauptfunktionen (GPIO + 3 Alternativfunktionen)

### Pin Multiplexing

Die meisten Portpins haben mehrere Ein-/Ausgangsfunktionen. Neben der Grundfunktion (primary function) gibt es bis zu 3 Alternativfunktionen (alternate function). Über die zugeordneten Pin-Select-Register (PINSELx) wird über einen Multiplexer das Portpin mit der entsprechenden On-Chip-Peripherie verbunden. Standardmäßig (nach Reset) sind die Pins mit den GPIO verbunden.

Beispiel Pin P0[0]:



I/O	<b>P0[0]</b> — General purpose digital input/output pin.
I	<b>RD1</b> — CAN1 receiver input.
O	<b>TXD3</b> — Transmitter output for UART3.

I/O **SDA1** — I<sup>2</sup>C1 data input/output (this pin does not use a specialized I<sup>2</sup>C pad, see [Section 19.1](#) for details).

### Basiskonfiguration Pins

#### a) PINCON Block

- |                                      |                            |                              |
|--------------------------------------|----------------------------|------------------------------|
| a. Pinfunktion auswählen             | Register <b>PINSEL</b>     | (GPIO nach Reset)            |
| Bei Bedarf:                          |                            |                              |
| b. Pull up/down Widerstand auswählen | Register <b>PINMODE</b>    | (bei Input; ON nach Reset)   |
| c. Open Drain Funktion auswählen     | Register <b>PINMODE_OD</b> | (bei Output; OFF nach Reset) |

#### b) GPIO Block

- |                           |                        |                    |
|---------------------------|------------------------|--------------------|
| a. Richtung I/O auswählen | Register <b>FIODIR</b> | (INPUT nach Reset) |
|---------------------------|------------------------|--------------------|

### Port-Ausgabe GPIO

- a) Pegel setzen
- b) Pegel löschen
- c) Pegel setzen oder löschen

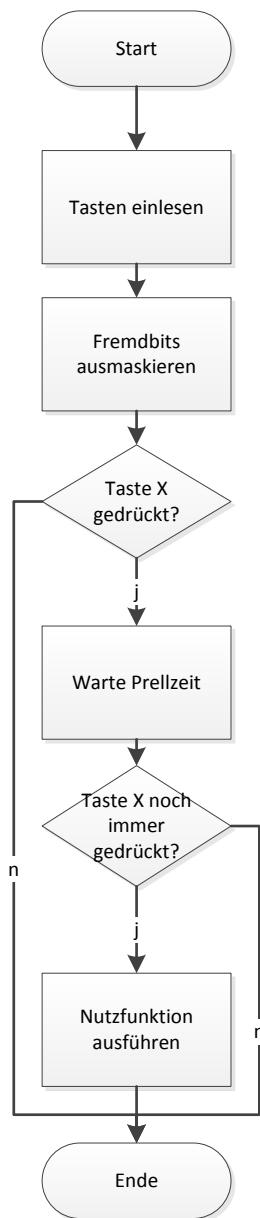
- |                        |                |
|------------------------|----------------|
| Register <b>FIOSET</b> | (0 nach Reset) |
| Register <b>FIOCLR</b> | (0 nach Reset) |
| Register <b>FIOPIN</b> | (0 nach Reset) |

## Tasten einlesen

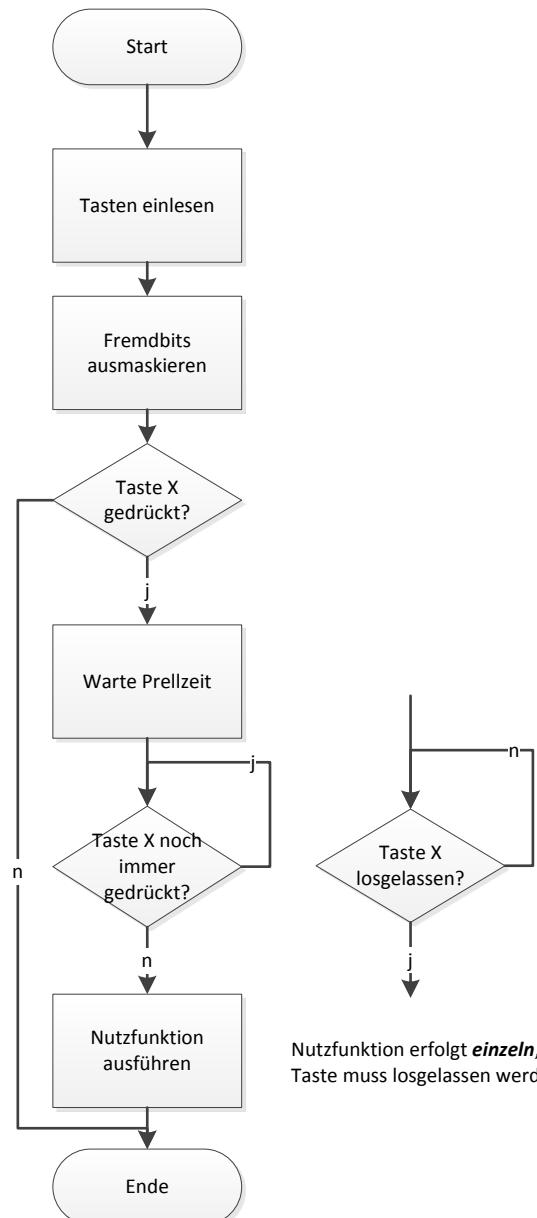
- a) Pegel feststellen (0 oder 1)
- b) Portpin abfragen
- c) Fremde Bits ausmaskieren
- d) Prellzeit beachten

Taste **nicht** gedrückt = ?      Taste gedrückt = ?  
**Register FIOPIN**  
**UND (AND)** Verknüpfung  
typisch <50ms (Datenblatt Schalter/Taste)

### TASTE EINLESEN Nutzfunktion ständig ausgeführt



### TASTE EINLESEN Nutzfunktion einzeln ausgeführt



#### Kopfgeprüfte Schleife:

```
while (Bedingung) {Nutzfunktion}
oder
if (Bedingung) {Nutzfunktion} else ...
```

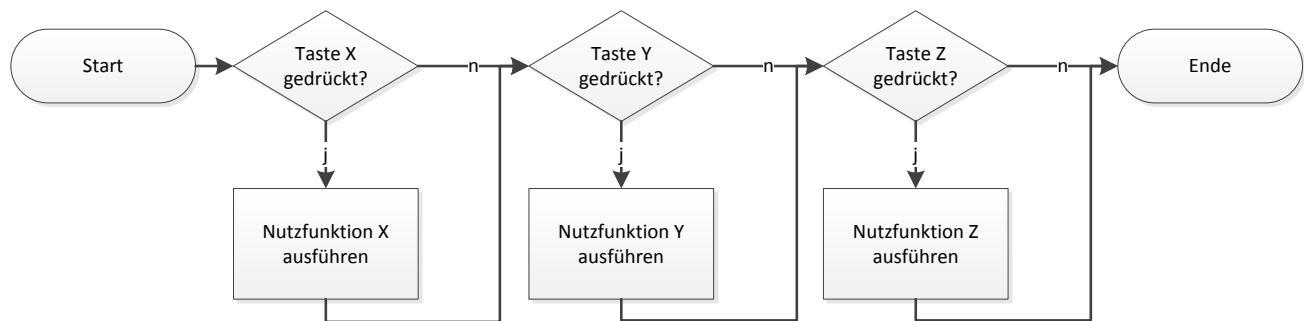
#### End/Fußgeprüfte Schleife:

```
do {Nutzfunktion} while (Bedingung);
verkürzte Variante:
while (Bedingung); //Semikolon zwingend!
```

### Mehrere Tasten einlesen, pseudo-gleichzeitig

Mehrere Tastenfunktionen die „pseudo“- gleichzeitig erfolgen müssen nacheinander abgearbeitet werden. Bei der Verwendung von Statusvariablen für den jeweiligen Tastenzustand sollten für jede Taste getrennt Variablen gewählt werden. Hier können nacheinander folgende if- Strukturen eingesetzt werden.

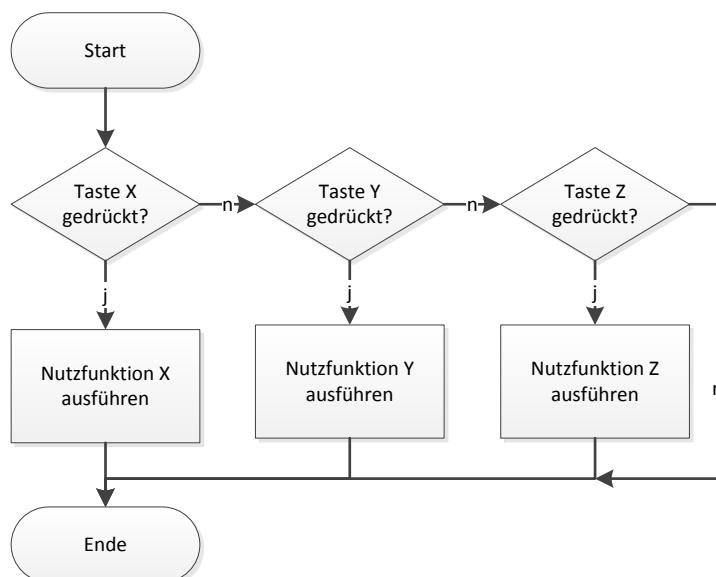
#### mehrere Tasten einlesen, pseudo-gleichzeitig (Prinzip)



### Mehrere Tasten einlesen, entweder/oder

Tastenfunktionen nach dem „entweder/oder“- Prinzip können durch verschachtelte if- Strukturen umgesetzt werden. Bei der Verwendung von Statusvariablen kann eine Variable für alle vorkommenden Tastenstati gewählt werden.

#### mehrere Tasten einlesen, entweder/oder (Prinzip)

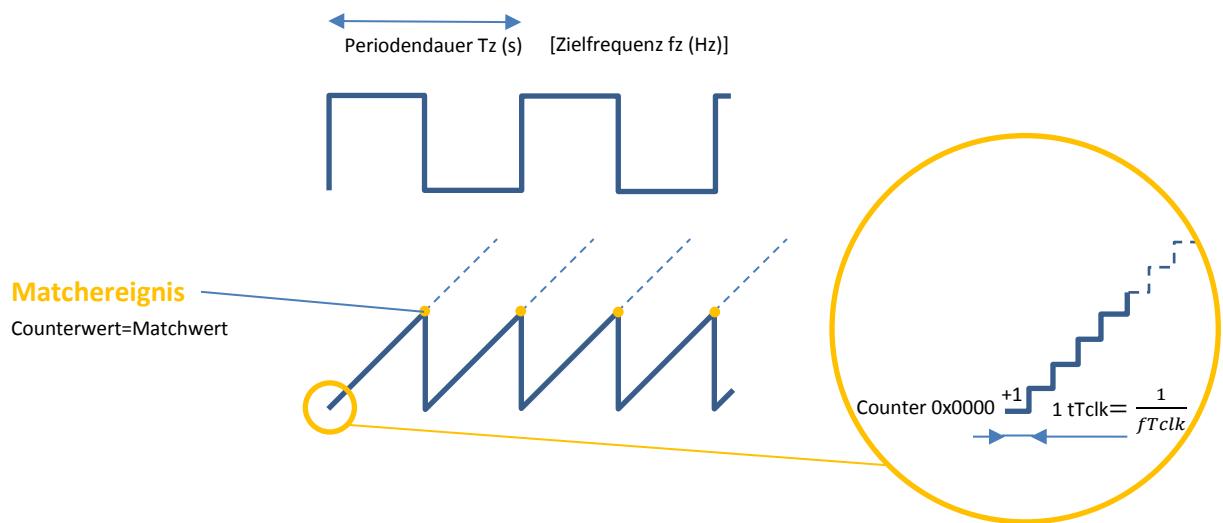


## 7. Timer (Kap. 21 Datenblatt)

- 4 Timer
- 32 Bit Zählregister
- Timer oder Counter (Zeitgeber oder Zähler)
- Capture/ Compare (Match) Funktionen
- Abschaltbar (Power-Off)
- U.a.

### Funktionsprinzip:

Bsp. Frequenzerzeugung



$$\text{Matchwert} = \frac{T_z}{tT_{clk}} \quad \text{Matchwert} = \frac{fT_{clk}}{2f_z} \quad \text{Matchwert} = fT_{clk} \times \frac{T_z}{2}$$

**T<sub>z</sub>:** Periodendauer  
Zielfrequenz  
**t<sub>Tclk</sub>:** Taktzeit Timer

**f<sub>z</sub>:** Zielfrequenz  
**f<sub>Tclk</sub>:** Timer-Frequenz

**T<sub>z</sub>:** Periodendauer  
Zielfrequenz  
**f<sub>Tclk</sub>:** Timer-Frequenz

## Blockbild Timer:

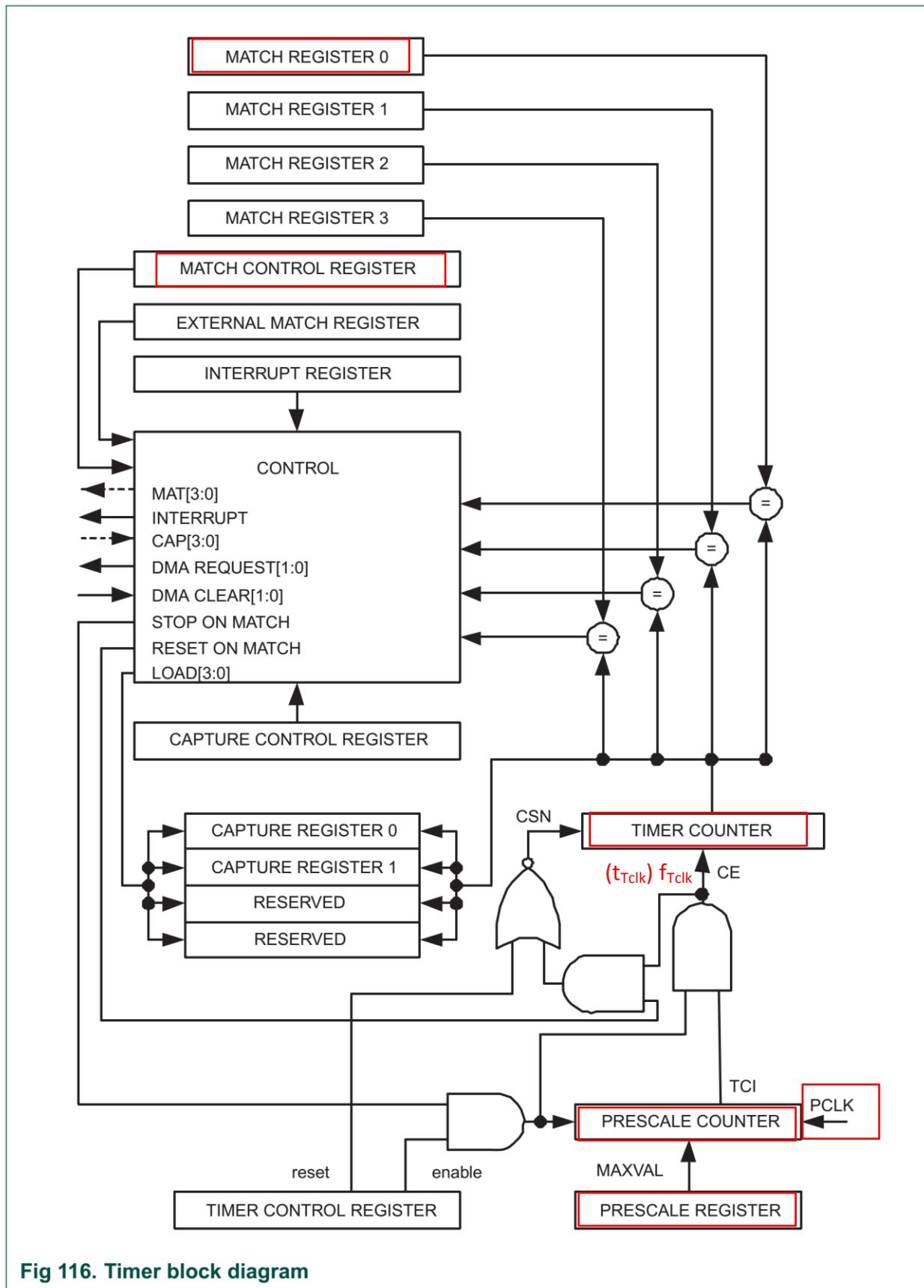


Fig 116. Timer block diagram

## Einstellung Timerfrequenz/ Timertakt :

Das Timer Counter Register wird nach jedem Timertakt um 1 inkrementiert. Der Timertakt bestimmt damit die Zeitdauer für 1 Inkrement. Frequenzbestimmend für den Timertakt ist der zugeordnete Prescale Counter sowie der Peripheral Clock (PCLK). Nach Reset ist der Prescale Counter aus (disable) und der Peripheral Clock  $\frac{1}{4}$  des CPU-Clocks.

### Timer Power On (PCONP, Kap. 4.8.9 Datenblatt)

Die Timer können per PCONP Register ein- bzw. ausgeschaltet werden. Timer 0 und 1 sind nach Reset eingeschaltet.

### Peripheral Clock (PCLK, Kap. 4.7.3 Datenblatt)

Der CPU-Clock kann durch das zugeordnete Peripheral-Clock-Select-Register (PCLKSELx) vorgeteilt werden und wird dann über den Peripheral-Clock (PCLK) bereitgestellt. Folgende Vorteilungen sind möglich:

#### Register PCLKSELx

1	Bitwert	01	PCLKSELx
2	Bitwert	10	PCLKSELx
4	Bitwert	00	PCLKSELx (nach Reset)
8	Bitwert	11	PCLKSELx

Der CPU-Clock beträgt 100MHz. Nach Reset ist damit der Peripheral Clock bei 25MHz.

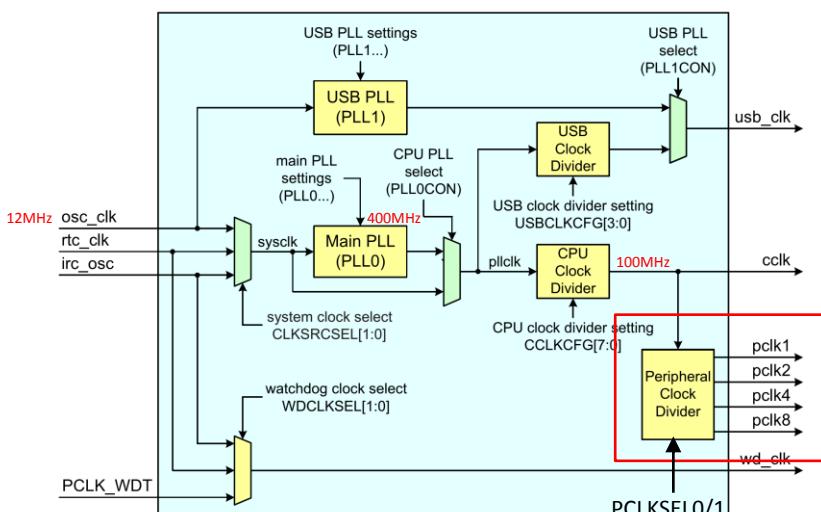


Fig 7. Clock generation for the LPC17xx

### Prescale Counter Register (TxPC, Kap. 21.6.6)

Der eingehende PCLK wird durch einen Wert geteilt. Sein Ausgangstakt wird dem Timer zugeführt ( $f_{TCLK}, t_{CLK}$ ). Das Prescale Counter Register inkrementiert bei jedem PCLK. Ist der Zählerwert gleich dem Wert im Prescale Register TxPR wird ein Timertakt  $t_{CLK}$  ausgelöst und das Prescale Counter Register rückgesetzt.

## Prescale Register (TxPR, Kap. 21.6.5)

Das Prescale Register enthält den Vergleichswert für den Prescale Counter.

**Wert Prescale Register** = gewünschter Teilungsfaktor-1

TxPR = 0	Timer Counter inkrementiert bei jedem PCLK = Faktor 1 (nach Reset)
TxPR = 1	... inkrementiert alle 2 PCLK (Faktor 2)
TxPR = 2	... inkrementiert alle 3 PCLK (Faktor 3) usw.

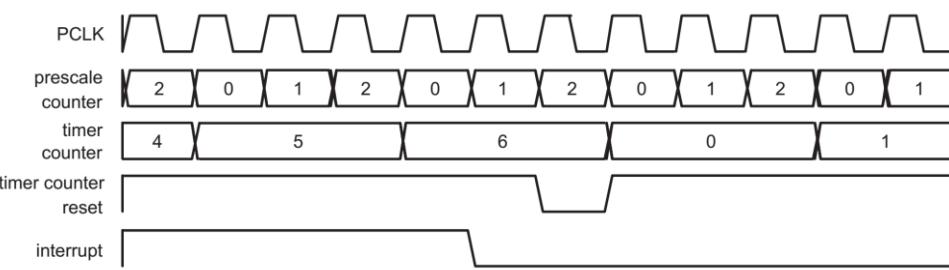


Fig 114. A timer cycle in which PR=2, MRx=6, and both interrupt and reset on match are enabled.

## Timer Modus (Kap. 21.6.3)

In Abhängigkeit der Einstellungen des Timer/Counter Control Registers (TxCTCR) arbeitet der Timer im Modus Zeitgeber (Timer) oder Zähler (Counter).

TxCTCR Bit[1:0]	Timer/ Counter Mode		Bitwert
	<b>Timer</b>	TC is incremented when the Prescale Counter matches the Prescale Register. The Prescale Counter is incremented on every rising PCLK edge.	<b>00</b>
	<b>Counter</b>	TC is incremented on rising edges on the Capture input CAP	<b>01</b>
	<b>Counter</b>	TC is incremented on falling edges on the Capture input CAP	<b>10</b>
	<b>Counter</b>	TC is incremented on both edges on the Capture input CAP	<b>11</b>

Nach Reset ist immer Timer-Mode und damit der Prescale Counter aktiv.

## Matchereignis (Kap. 21.6.7)

Ist der Inhalt des **Timercounter Registers = Inhalt Matchregister**, kann ein bestimmtes Ereignis ausgelöst werden.

Der LPC1768 besitzt für jeden Timer

4 Match-Register, die Matchregister 0-3.



Für **einen** Timer können **4 verschiedene** Ereignisse ausgelöst werden.

## Match Control Register

Konfiguriert die Aktionen, die bei Matchereignis durchgeführt werden.

Die Steuerung dafür erfolgt über jeweils 3 Bits im zugewiesenen **Match Control Register TxMCR**.

- MRxI = 1      **INTERRUPT** auslösen
- MRxR = 1      **RESET** Timer Counter (TC)=0000 0000
- MRxS = 1      **STOP** TC=halt, PC=halt, TCCR.0=0

**Table 429. Match Control Register (T[0/1/2/3]MCR - addresses 0x4000 4014, 0x4000 8014, 0x4009 0014, 0x4009 4014)**  
bit description

Bit	Symbol	Value	Description	Reset Value
0	MR0I	1	Interrupt on MR0: an interrupt is generated when MR0 matches the value in the TC.	0
		0	This interrupt is disabled	
1	MR0R	1	Reset on MR0: the TC will be reset if MR0 matches it.	0
		0	Feature disabled.	
2	MR0S	1	Stop on MR0: the TC and PC will be stopped and TCR[0] will be set to 0 if MR0 matches the TC.	0
		0	Feature disabled.	
3	MR1I	1	Interrupt on MR1: an interrupt is generated when MR1 matches the value in the TC.	0
		0	This interrupt is disabled	
4	MR1R	1	Reset on MR1: the TC will be reset if MR1 matches it.	0
		0	Feature disabled.	
5	MR1S	1	Stop on MR1: the TC and PC will be stopped and TCR[0] will be set to 0 if MR1 matches the TC.	0
		0	Feature disabled.	
6	MR2I	1	Interrupt on MR2: an interrupt is generated when MR2 matches the value in the TC.	0
		0	This interrupt is disabled	
7	MR2R	1	Reset on MR2: the TC will be reset if MR2 matches it.	0
		0	Feature disabled.	
8	MR2S	1	Stop on MR2: the TC and PC will be stopped and TCR[0] will be set to 0 if MR2 matches the TC.	0
		0	Feature disabled.	
9	MR3I	1	Interrupt on MR3: an interrupt is generated when MR3 matches the value in the TC.	0
		0	This interrupt is disabled	
10	MR3R	1	Reset on MR3: the TC will be reset if MR3 matches it.	0
		0	Feature disabled.	
11	MR3S	1	Stop on MR3: the TC and PC will be stopped and TCR[0] will be set to 0 if MR3 matches the TC.	0
		0	Feature disabled.	
31:12 -			Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

## External Match - Pinsteuerung durch Matchereignis (Kap.21.6.11):

Bei einem Matchereignis kann ein Pegelwechsel an einem Portpin erfolgen. Diese alternative Portfunktion ist durch Register **PINSELx** zu aktivieren. Der ausgewiesene Kanal (Channel\_x) entspricht dabei dem gewählten Matchregister MRx des Timers. Die Konfiguration erfolgt durch das Register **TxEMR**.

### Register TxEMR (External Match Register)

Das Register enthält die Information für Steuerung und Status der externen Portpins. Jedem Matchregister ist 1 Portpin zugeordnet.

Status: Bits 0 – 3 = Zustand des Portpins (positive Logik, 1=High, 0=Low)

Steuerung (Control): Bits 4 – 11

- 00 nichts tun
- 01 Status/Portausgang löschen = 0
- 10 Status/Portausgang setzen = 1
- 11 Status/Portausgang invertieren(toggeln) = 0 oder 1

**Table 431. External Match Register (T[0/1/2/3]EMR - addresses 0x4000 403C, 0x4000 803C, 0x4009 003C, 0x4009 403C) bit description**

Bit	Symbol	Description	Reset Value
0	EM0	External Match 0. When a match occurs between the TC and MR0, this bit can either toggle, go low, go high, or do nothing, depending on bits 5:4 of this register. This bit can be driven onto a MATn.0 pin, in a positive-logic manner (0 = low, 1 = high).	0
1	EM1	External Match 1. When a match occurs between the TC and MR1, this bit can either toggle, go low, go high, or do nothing, depending on bits 7:6 of this register. This bit can be driven onto a MATn.1 pin, in a positive-logic manner (0 = low, 1 = high).	0
2	EM2	External Match 2. When a match occurs between the TC and MR2, this bit can either toggle, go low, go high, or do nothing, depending on bits 9:8 of this register. This bit can be driven onto a MATn.2 pin, in a positive-logic manner (0 = low, 1 = high).	0
3	EM3	External Match 3. When a match occurs between the TC and MR3, this bit can either toggle, go low, go high, or do nothing, depending on bits 11:10 of this register. This bit can be driven onto a MATn.3 pin, in a positive-logic manner (0 = low, 1 = high).	0
5:4	EMC0	External Match Control 0. Determines the functionality of External Match 0. <a href="#">Table 432</a> shows the encoding of these bits.	00
7:6	EMC1	External Match Control 1. Determines the functionality of External Match 1. <a href="#">Table 432</a> shows the encoding of these bits.	00
9:8	EMC2	External Match Control 2. Determines the functionality of External Match 2. <a href="#">Table 432</a> shows the encoding of these bits.	00
11:10	EMC3	External Match Control 3. Determines the functionality of External Match 3. <a href="#">Table 432</a> shows the encoding of these bits.	00
15:12 -		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

**Table 432. External Match Control**

EMR[11:10], EMR[9:8], Function EMR[7:6], or EMR[5:4]	
00	Do Nothing.
01	Clear the corresponding External Match bit/output to 0 (MATn.m pin is LOW if pinned out).
10	Set the corresponding External Match bit/output to 1 (MATn.m pin is HIGH if pinned out).
11	Toggle the corresponding External Match bit/output.

## Timer starten/rücksetzen (Kap. 21.6.2)

### Timer Control Register TxTCR

Bit 0 TCR = 1	Timer start (enable)
Bit 0 TCR = 0	Timer stop (disable) = Zustand nach Reset
Bit 1 TCR = 1	Reset: TC=0000 0000, PC=0000 0000 (solange Bit =1)
Bit 1 TCR = 0	(Freigabe Zählen)

### Basiskonfiguration:

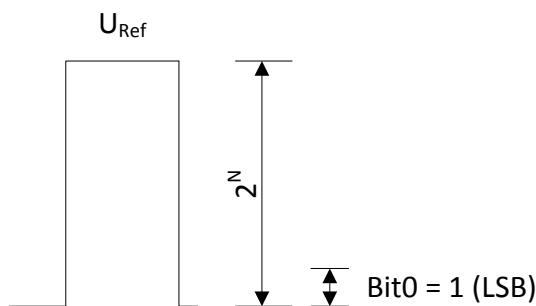
- |   |  |
|---|--|
| c) Power On   | Register <b>PCONF</b> (Timer 0 und 1 nach Reset On)                    |
| d) Peripheral Clock Timer                           | Register <b>PCLKSEL0/1</b> (1/4 CPUCLK nach Reset)                     |
| e) Match Ereignis Steuerung<br>(Vergleichsereignis) | Register <b>TxMCR</b><br>(Interrupt auslösen/ Timer Reset/ Timer Stop) |
| f) Matchregister laden<br>(Vergleichswert)          | Register <b>MRx</b>  |
| g) External Match Steuerung                         | (Pinsteuerung, bei Bedarf, Register <b>EMR</b> , Bits <b>EMCx</b> )    |
| h) Alternative Pinfunktionen?                       | Register <b>PINSELx</b> (bei Bedarf z.B. external Match)               |
| i) Interrupt freigeben                              | (bei Bedarf, Interruptpriorität und Freigabe einstellen)               |
| j) Timer starten                                    | Register <b>TxTCR</b>  |

## 8. ADC (Analog Digital Wandler)

- 12-Bit ADC (max. FFF; 4096 Zustände)
- Verfahren successive approximation
- 8 Kanäle (8 Pins)
- Wandlungsraten max. 200kHz
- Power Down Mode
- U.a.

### Grundlagen:

Zählumfang eines Registers ( $2^N$ ) wird auf eine Vergleichsspannung (Referenzspannung  $U_{Ref}$ ) verteilt.



$$U_{LSB} = \frac{U_{Ref}}{2^N}$$

$$\text{Digitalwert ADC-Register} = \frac{\text{Eingangsspannung ADC}}{U_{LSB}}$$

### Modul LPC1768:

ADC-Eingang an PinX.x

$U_{Ref} = 3,3V$

Auflösung ADC = 12 Bit

$$U_{LSB} = \frac{3,3V}{4096} = 0,806 \text{ mV}$$

### Basiskonfiguration:

- |                                    |  |                            |
|------------------------------------|--|----------------------------|
| a. Power On                        | Register <b>PCONP</b> , Bit <b>PCADC</b> | (nach Reset Off)           |
| b. Peripheral Clock ADC einstellen | Register <b>PCLKSEL0</b>                 | (1/4 CPUCLK nach Reset)    |
| c. Pinfunktionen für ADC auswählen | Register <b>PINSEL0..n</b> ,             | (evtl. PINMODE bei Bedarf) |
| d. ADC Modi konfigurieren          | Register <b>ADxCR</b>                    |                            |
| e. ADC kalibrieren (trimmen)       | (bei Bedarf Register <b>ADTRM</b> )      |                            |
| f. Interrupt freigeben             | (bei Bedarf, Register <b>ADxINTEN</b> )  |                            |
| g. DMA freigeben                   | (nur bei Bedarf)                         |                            |
| h. ADC starten                     | Register <b>ADxCR, PDN=1</b>             |                            |

## Datenregister

### ADDR0 bis ADDR7

enthalten den gewandelten Binärwert des Kanals und zugehörige Statusflags

### ADGDR

globales Datenregister, enthält letzten gewandelten Wert sowie dazugehörige Statusflags

## Statusregister

### ADSTAT

enthält Statusflags zur Wandlung

## Konfigurationsregister ADxCR

### (Kap. 29.5.1, S. 577 Datenblatt)

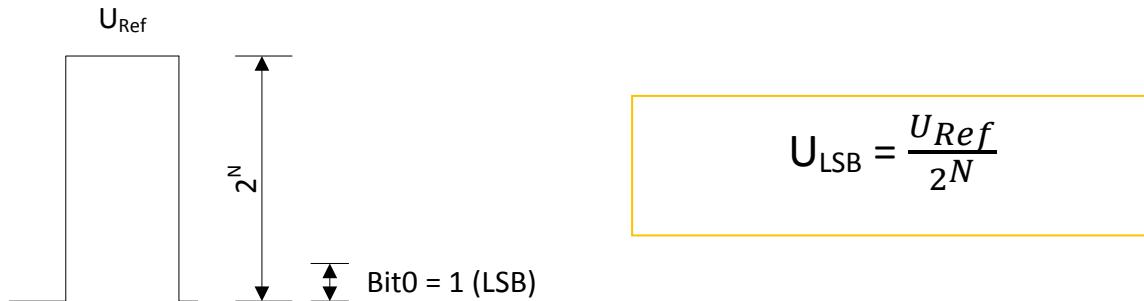
Bei LPC1767 nur Register AD0CR verfügbar.

Folgende Einstellungen werden konfiguriert:

Bits	Symbol	Aufgabe	Funktion, wenn gesetzt (1)
Bit 0-7	SEL	Kanalauswahl	Kanälen, in denen Wandlungen durchgeführt werden sollen
Bit 8-15	CLKDIV	ADC-Takt festlegen	Takt=PCLK_ADC0/CLKDIV, sollte im Ergebnis knapp unter 13MHz liegen, Zahl zwischen 0-255
Bit 16	BURST	REPEAT-MODE	<b>Automatisch ständige Wandlung auf allen gewählten Kanälen, beginnend bei Kanal0; (Bit 24-26 START müssen 000 sein!)</b>
Bit 21	PDN	Betrieb/ Power-Down	Betrieb = 1 Power-Down=0 (nach Reset 0)
Bit 24-26	START	SINGLE-MODE	<b>Einzelwandlung, kann durch verschiedene Ereignisse ausgelöst werden (Bit 16 BURST muss 0 sein!)</b>
(Bit 27)	EDGE	Flankenauswahl (bei Bedarf)	reagiert auf fallende Flanke Signal CAP/MAT

## 9. Digital-Analog-Wandler DAC

Zählumfang eines Registers ( $2^N$ ) wird auf eine Vergleichsspannung (Referenzspannung  $U_{Ref}$ ) verteilt.



### Modul LPC1768:

$U_{Ref} = 3,3V$ , Auflösung DAC = 10 Bit

DAC-Ausgang an Pin

$$U_{LSB} = \frac{3,3V}{1024} = 3,223 \text{ mV}$$

$$\text{Ausgangsspannung DAC} = U_{LSB} \times \text{Digitalwert}$$

$$\text{Digitalwert DAC} = \frac{\text{Ausgangsspannung DAC}}{U_{LSB}}$$

### Basiskonfiguration:

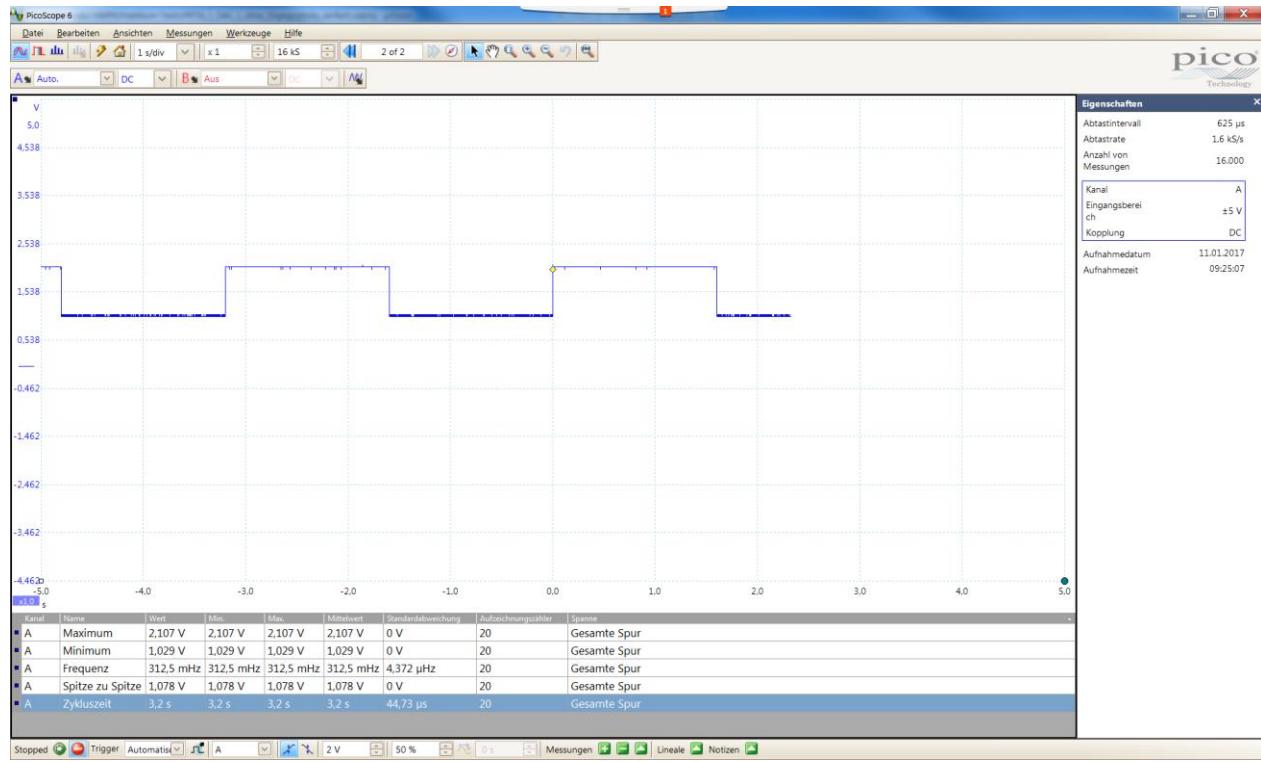
#### 30.1 Basic configuration

The DAC is configured using the following registers:

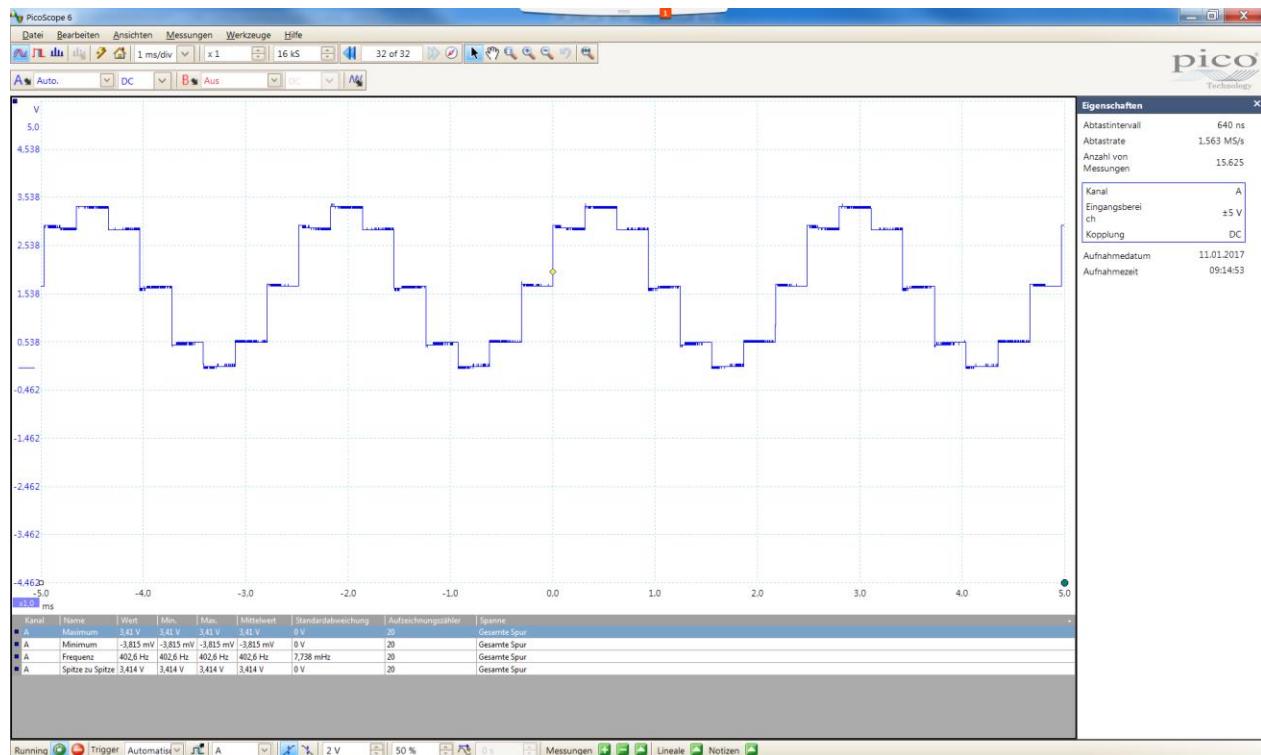
1. Power: The DAC is always connected to  $V_{DDA}$ . Register access is determined by PINSEL and PINMODE settings (see below).
2. Clock: In the PCLKSEL0 register ([Table 40](#)), select PCLK\_DAC.
3. Pins: Enable the DAC pin through the PINSEL registers. Select pin mode for port pin with DAC through the PINMODE registers ([Section 8.5](#)). This must be done before accessing any DAC registers.
4. DMA: The DAC can be connected to the GPDMA controller (see [Section 30.4.2](#)). For GPDMA connections, see [Table 543](#).

## Verwendung des Software-Ozilloskops „PicoScope“

- Anschluss am **USB-Anschluss des PC** (darüber erfolgt auch die Stromversorgung)
- Start **Programm PicoScope 6** (Programme->Pico Technology->PicoSope6)
- Anschluss des **Tastkopfes** (Messspitze) an **Jumper 3** (Seite zur Batterie), **Masseclip** an **Gehäuse SD-Karte**



Ausgabe Projekt 7\_1\_DAC\_1\_ohne\_Display



Ausgabe Projekt 7\_3\_DAC\_3\_sinus\_1

## 10. Display

Verwendet wird das Standarddisplay **GDM2004D**.

(Datenblatt R:\CB\Bader\PR HWPR\Dokumente Hardware\4\_DISPLAY\_gdm2004d\_datasheet.pdf)

### Eigenschaften:

- 4x20 Zeichen (4 Zeilen, 20 Spalten)
- 5x8 Punkte je Zeichen (Punktmatrix)
- Befehlsgesteuert
- 8 Bit oder 4 Bit Datenübertragungsmodus
- 3 Steuersignale (RS, R/W, E)
- Kontrastregelung (V0)

### Einsatz auf TestBed:

- 4 Bit Übertragungsmodus
- Hintergrundbeleuchtung nicht steuerbar
- Kontrastregelung über Potentiometer
- Signalzuordnung:

Funktion	LCD-Anschluss	mbed-Signal
<i>GND</i>	1	-
<i>5V</i>	2	-
<i>Contrast</i>	3	-
<i>RS</i>	4	P26
<i>R/W</i>	5	P25
<i>EN</i>	6	P24
<i>DB0</i>	7	-
<i>DB1</i>	8	-
<i>DB2</i>	9	-
<i>DB3</i>	10	-
<i>DB4</i>	11	P23
<i>DB5</i>	12	P22
<i>DB6</i>	13	P20
<i>DB7</i>	14	P19
<i>Backlight+</i>	15	-
<i>Backlight-</i>	16	-

### Software- Bibliothek: **lcd\_4bit\_hd44780.c**

#### Funktionen

void lcd_init(void)	Display initialisieren
void lcd_putchar(unsigned char)	1 Zeichen senden
void lcd_putstr(unsigned char*)	Zeichenkette senden
void lcd_write_number(int)	Zahl senden
void lcd_cursor(int,int)	Cursor setzen (X, Y)
void delay_ms(int)	warte ? ms
void delay_us(int);	warte ? $\mu$ s
void lcd_clear(void);	Display löschen



## ASCII-Zeichentabelle

Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr		Dec	Hx	Oct	Html	Chr		Dec	Hx	Oct	Html	Chr
0	0 000	000	<b>NUL</b> (null)		32	20 040	000	&#32;	<b>Space</b>		64	40 100	000	&#64;	<b>Ø</b>		96	60 140	000	&#96;	'
1	1 001	001	<b>SOH</b> (start of heading)		33	21 041	001	&#33;	!		65	41 101	001	&#65;	<b>A</b>		97	61 141	001	&#97;	<b>a</b>
2	2 002	002	<b>STX</b> (start of text)		34	22 042	002	&#34;	"		66	42 102	002	&#66;	<b>B</b>		98	62 142	002	&#98;	<b>b</b>
3	3 003	003	<b>ETX</b> (end of text)		35	23 043	003	&#35;	#		67	43 103	003	&#67;	<b>C</b>		99	63 143	003	&#99;	<b>c</b>
4	4 004	004	<b>EOT</b> (end of transmission)		36	24 044	004	&#36;	\$		68	44 104	004	&#68;	<b>D</b>		100	64 144	004	&#100;	<b>d</b>
5	5 005	005	<b>ENQ</b> (enquiry)		37	25 045	005	&#37;	%		69	45 105	005	&#69;	<b>E</b>		101	65 145	005	&#101;	<b>e</b>
6	6 006	006	<b>ACK</b> (acknowledge)		38	26 046	006	&#38;	&		70	46 106	006	&#70;	<b>F</b>		102	66 146	006	&#102;	<b>f</b>
7	7 007	007	<b>BEL</b> (bell)		39	27 047	007	&#39;	!		71	47 107	007	&#71;	<b>G</b>		103	67 147	007	&#103;	<b>g</b>
8	8 010	010	<b>BS</b> (backspace)		40	28 050	010	&#40;	(		72	48 110	010	&#72;	<b>H</b>		104	68 150	010	&#104;	<b>h</b>
9	9 011	011	<b>TAB</b> (horizontal tab)		41	29 051	011	&#41;	)		73	49 111	011	&#73;	<b>I</b>		105	69 151	011	&#105;	<b>i</b>
10	A 012	012	<b>LF</b> (NL line feed, new line)		42	2A 052	012	&#42;	*		74	4A 112	012	&#74;	<b>J</b>		106	6A 152	012	&#106;	<b>j</b>
11	B 013	013	<b>VT</b> (vertical tab)		43	2B 053	013	&#43;	+		75	4B 113	013	&#75;	<b>K</b>		107	6B 153	013	&#107;	<b>k</b>
12	C 014	014	<b>FF</b> (NP form feed, new page)		44	2C 054	014	&#44;	,		76	4C 114	014	&#76;	<b>L</b>		108	6C 154	014	&#108;	<b>l</b>
13	D 015	015	<b>CR</b> (carriage return)		45	2D 055	015	&#45;	-		77	4D 115	015	&#77;	<b>M</b>		109	6D 155	015	&#109;	<b>m</b>
14	E 016	016	<b>SO</b> (shift out)		46	2E 056	016	&#46;	.		78	4E 116	016	&#78;	<b>N</b>		110	6E 156	016	&#110;	<b>n</b>
15	F 017	017	<b>SI</b> (shift in)		47	2F 057	017	&#47;	/		79	4F 117	017	&#79;	<b>O</b>		111	6F 157	017	&#111;	<b>o</b>
16	10 020	020	<b>DLE</b> (data link escape)		48	30 060	020	&#48;	0		80	50 120	020	&#80;	<b>P</b>		112	70 160	020	&#112;	<b>p</b>
17	11 021	021	<b>DC1</b> (device control 1)		49	31 061	021	&#49;	1		81	51 121	021	&#81;	<b>Q</b>		113	71 161	021	&#113;	<b>q</b>
18	12 022	022	<b>DC2</b> (device control 2)		50	32 062	022	&#50;	2		82	52 122	022	&#82;	<b>R</b>		114	72 162	022	&#114;	<b>r</b>
19	13 023	023	<b>DC3</b> (device control 3)		51	33 063	023	&#51;	3		83	53 123	023	&#83;	<b>S</b>		115	73 163	023	&#115;	<b>s</b>
20	14 024	024	<b>DC4</b> (device control 4)		52	34 064	024	&#52;	4		84	54 124	024	&#84;	<b>T</b>		116	74 164	024	&#116;	<b>t</b>
21	15 025	025	<b>NAK</b> (negative acknowledge)		53	35 065	025	&#53;	5		85	55 125	025	&#85;	<b>U</b>		117	75 165	025	&#117;	<b>u</b>
22	16 026	026	<b>SYN</b> (synchronous idle)		54	36 066	026	&#54;	6		86	56 126	026	&#86;	<b>V</b>		118	76 166	026	&#118;	<b>v</b>
23	17 027	027	<b>ETB</b> (end of trans. block)		55	37 067	027	&#55;	7		87	57 127	027	&#87;	<b>W</b>		119	77 167	027	&#119;	<b>w</b>
24	18 030	030	<b>CAN</b> (cancel)		56	38 070	030	&#56;	8		88	58 130	030	&#88;	<b>X</b>		120	78 170	030	&#120;	<b>x</b>
25	19 031	031	<b>EM</b> (end of medium)		57	39 071	031	&#57;	9		89	59 131	031	&#89;	<b>Y</b>		121	79 171	031	&#121;	<b>y</b>
26	1A 032	032	<b>SUB</b> (substitute)		58	3A 072	032	&#58;	:		90	5A 132	032	&#90;	<b>Z</b>		122	7A 172	032	&#122;	<b>z</b>
27	1B 033	033	<b>ESC</b> (escape)		59	3B 073	033	&#59;	:		91	5B 133	033	&#91;	[		123	7B 173	033	&#123;	{
28	1C 034	034	<b>FS</b> (file separator)		60	3C 074	034	&#60;	<		92	5C 134	034	&#92;	\		124	7C 174	034	&#124;	
29	1D 035	035	<b>GS</b> (group separator)		61	3D 075	035	&#61;	=		93	5D 135	035	&#93;	]		125	7D 175	035	&#125;	}
30	1E 036	036	<b>RS</b> (record separator)		62	3E 076	036	&#62;	>		94	5E 136	036	&#94;	^		126	7E 176	036	&#126;	~
31	1F 037	037	<b>US</b> (unit separator)		63	3F 077	037	&#63;	?		95	5F 137	037	&#95;	_		127	7F 177	037	&#127;	<b>DEL</b>

Source: [www.LookupTables.com](http://www.LookupTables.com)