

Einleitung

DMXLib ist eine Sammlung von Routinen die den Empfang und die Auswertung von DMX Daten via MCS 51 kompatiblen Mikrocontrollern. DMXLib setzt einen 8051 Controller mit 16 MHz Taktfrequenz voraus. Diese Version unterstützt 4 Schaltausgänge konfigurierbar zwischen DMX Kanal 0 und DMX Kanal 255. Für die Entwicklung wurde der Freeware C Compiler SDCC von Sandeep Dutta verwendet.

Verwendung der Ressourcen des 8051

Die Bibliothek beansprucht folgende Ressourcen des Controllers.

1. Einen Datenbuffer von 4 Byte für die einzelnen DMX Ausgänge
2. 4 Byte für globale Variablen
3. 6 Bit für Flaggen
4. Timer 1 für Timeout
5. UART für DMX Empfang
6. Port 2.3 bis 2.4 als Schaltausgänge
7. Port 1 für 8 x DIP konfigurationsabhängig

Der UART und der Timer 1 dürfen in Projekten, die das DMX Library verwenden nicht benutzt werden, weil dadurch deren korrekte Funktion nicht mehr gewährleistet ist.

Allgemeine Funktionsbeschreibung

Der zentrale Bestandteil der DMX Bibliothek ist die Interrupt Funktion **UART**. Diese Funktion kontrolliert den Empfang von DMX Daten, sowie deren Pufferung im RAM des Controllers. Die Funktion UART kann mittels der Funktion SetDMX mit verschiedenen Parametern initialisiert werden.

In der jetzigen Version von DMXLib werden nur DMX Strings mit Startcode 0x0 verarbeitet. Alle anderen Datenstrings werden ignoriert. Grundsätzlich läuft der Empfang von DMX Strings wie folgt ab. Nach erfolgter Konfiguration der Empfänger-Eigenschaften wartet der Controller auf ein gültiges DMX Break. Ist ein Break erkannt, wird das nächste empfangene Datenbyte, das müsste dann der Startcode sein, überprüft. Ist dieser Startcode 0x0 wird die Abspeicherung der Daten eingeleitet. Je nach Konfiguration kann man die einzelnen Module als Teile einer komplexeren Software verwenden oder als selbständige Funktionseinheit. In letzterem Fall arbeitet die Software selbständig als Hintergrundtask und regelt alle Funktionen vom Empfang bis zur Auswertung der Daten.

Bibliotheksfunktionen

unsigned char GetDip()

Diese Funktion liest den 8 x DIP am Port P1 des Controllers bildet daraus die gültige DMX Adresse und gibt diese zurück. Gleichzeitig wird der DMX Kanal zur globalen Verwendung in der Bibliothek gespeichert. Der Adressbereich geht von 0 -255.

Anschluss des DIP am Controller

Belegung	
Portpin	Verwendung
P1.0	DIP1
P1.1	DIP2
P1.2	DIP3
P1.3	DIP4
P1.4	DIP5
P1.5	DIP6
P1.6	DIP7
P1.7	DIP8

Kanalbelegungstabelle

Mit Hilfe des 8 x DIP Schalters lassen sich folgende DMX Kanäle einstellen.

DIP				1															
				2															
				3															
				4															
8	7	6	5	DMX Kanäle															
				0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
				16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
				32	33	34	35	36	37	38	38	40	41	42	43	44	45	46	47
				48	49	50	51	52	53	53	55	56	57	58	59	60	51	52	63
				64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
				80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
				96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
				112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
				128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
				144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
				160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
				176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
				192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
				208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
				224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
				240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

void SetDMX(unsigned char Param)

Diese Funktion konfiguriert den DMX Empfang des Controllers. Sie initialisiert den Datenbuffer mit dem Wert 0 für jeden Ausgang, setzt den Timeoutwert auf 1 Sekunde und startet den DMX Empfang. Für den Aufruf werden einige Parameter verwendet, die folgende Bedeutung haben.

Param		
Bitnummer	Name	Funktion
0	OPTUseSwitch	automatische Auswertung der DMX Werte als Switch nach jedem korrekt Empfangenen DMX Rahmen
1	OPTUseDIP	automatische Verwendung des DIP Schalter zu Kanaleinstellung Empfangsroutine
2	OPTUseTh	Setzt den Schaltwert für die Routine Switch auf den Default Wert (128)
3	X	X
4	X	X
5	X	X
6	X	X
7	X	X

void StopDMX()

Diese Funktion stoppt den Empfang und die Auswertung von DMX Rahmen. Weiterhin löscht sie den internen Datenbuffer und inaktiviert die Ausgänge.

void Switch()

Diese Funktion wertet die eingegangenen Daten als Schalter aus, des heißt das beim Überschreiten des Wertes für die Schaltschwelle eines Kanals dessen Ausgang aktiviert wird. Die Ausgänge für die Schaltfunktion sind **LOW** aktiv d.h. das ein aktivierter Kanal dessen zugeordneten Ausgang auf 0 Volt liegt. Die Funktion kann manuell durch das Hauptprogramm aufgerufen werden, bzw. automatisch durch die Empfangsroutine, wenn beim Initialisieren des Empfängers mittels SetDMX das BIT OPTUseSwitch gesetzt wurde. Automatisch heißt das die Empfangsroutine selbständig die Funktion Switch Aufruft wenn ein korrekter DMX Rahmen Empfangen wurde. Die Funktion Switch wertet 4 Kanäle fortlaufend vom eingestellten DMX Kanal aus.

der Funktion Switch zugeordnete Ausgänge	
Port	
2.4	Ausgang 1
2.5	Ausgang 2
2.6	Ausgang 3
2.7	Ausgang 4

void SetThreshold (unsigned char thr)

Die Funktion SetThreshold setzt den Wert für die Schaltschwelle die durch die Funktion Switch verwendetet wird. Der Wert für die Schaltschwelle wird in der Variablen thr übergeben. Gültige Werte sind von 0-255.

unsigned char GetThreshold()

Die Funktion GetThreshold gibt den aktuell gesetzten Wert für die Schaltschwelle der Funktion Switch zurück.

void SetDMXChannel(unsigned char channel)

Die Funktion SetDMXChannel setzt den Wert des aktuellen DMX Kanals. Der Wert wird in der Variablen **channel** übergeben. Gültige Werte sind im Bereich von 0 – 255. Die Funktion sollte nur verwendet werden wenn die automatische DIP Abfrage deaktiviert ist, da sonst der neu gesetzte DMX Kanal von der Routine GetDip überschrieben wird.

unsigned char GetDMXChannel()

Die Funktion GetDMXChannel gibt den aktuell gesetzten DMX Kanal zurück.

unsigned char GetDMXValue(unsigned char index)

Die Funktion GetDMXValue gibt den durch die Variable Index indizierten Wert im Datenbuffer des Controllers zurück, d.h. eine übergebene 1 in der Variablen Index gibt den DMX Wert für den Gerätekanal DMX Kanal +1 zurück. Gültige Werte liegen im Bereich von 0 bis 3.

unsigned char GetDMXStatus()

Gibt den Status des DMX Receivers zurück. Eine 0 sagt aus, dass der DMX Empfang aktiv ist, während eine 255 aussagt dass ein DMX Timeout vorliegt.

Ein Beispielprojekt auf Basis der DMXLib

Im folgendem soll ein kleines Projekt die Anwendung von DMXLib demonstrieren. Dieses Projekt stellt einen kleinen 4 Kanal DMX Receiver dar, welcher zum Beispiel als 4 Kanal Switchpack oder 4 Kanal Relaisbank verwendet werden kann. In diesem Fall wird der DMX Receiver mit dem Wert 0x7 initialisiert. Das heißt dass der DMX Empfang selbständig quasi als Hintergrundtask läuft. Mit diesem einfachen Programm hat man schon einen vollständig funktionierenden DMX Empfänger.

Die Datei mini.c

```
#include "dmx.h"
```

```
void main()
```

```
{
```

```
SetDMX(0x7);
```

```
//
```

```
der DMX Receiver arbeitet
```

```
//
```

```
selbständig als Hintergrundtask
```

```
warte:
```

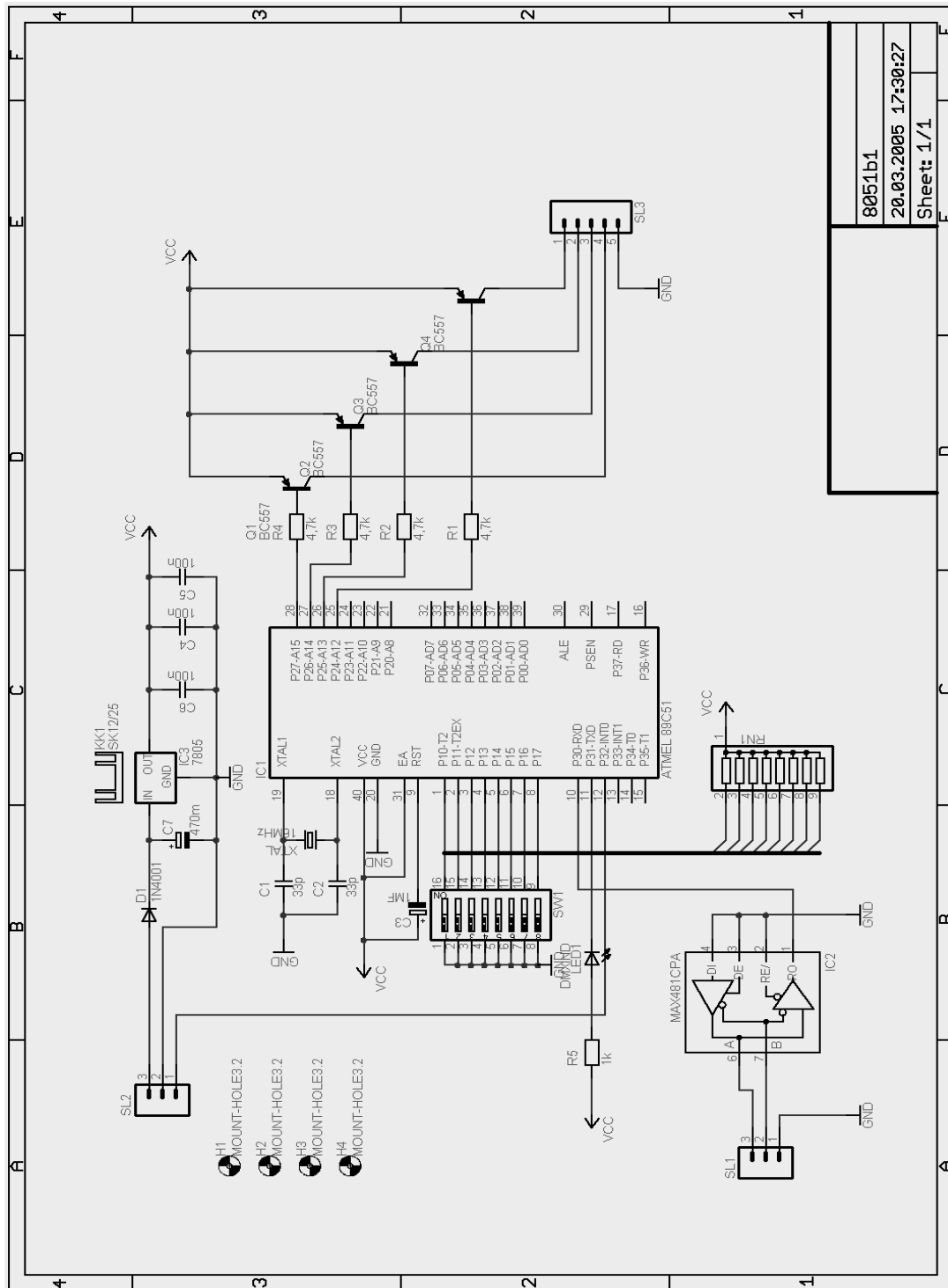
```
// mögliche eigene Funktionen
```

```
goto warte;
```

```
}
```

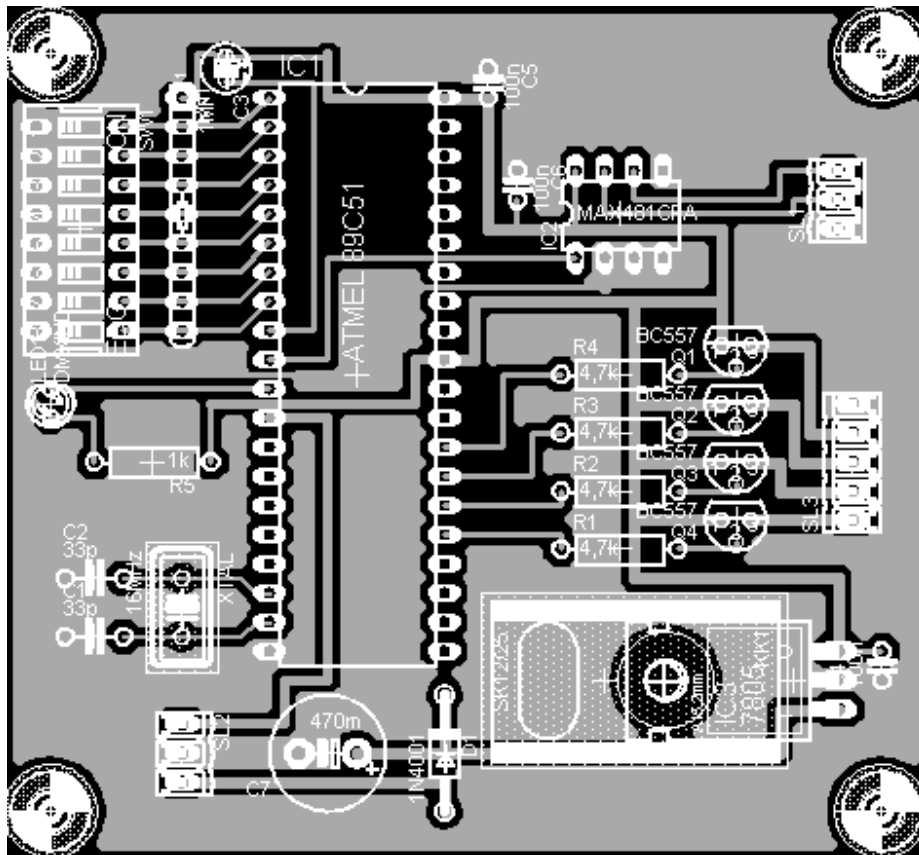
Das Projekt besteht aus folgenden Dateien **mini.c**, **dmxlib.c**, **dmxlib.h**, **ub.bat**. Die Datei **ub.bat** ist die Stapelverarbeitungsdatei mit deren Hilfe das Projekt compiliert wird. Die Compilierung des Projekts erzeugt die Datei **mini.ihx**. Diese Datei ist ein Intel Hex File und kann direkt in den Controller geschrieben werden.

Schaltplan eines 4 Kanal DMX Receivers



8051b1
20.03.2005 17:30:27
Sheet: 1/1

Platinenlayout des 4 Kanal DMX Receivers



Steckerbelegungen

Belegung SL1 DMX	
Pin	Verwendung
1	DMX +
2	DMX -
3	GND

Belegung SL2 POWER	
Pin	Verwendung
1	INT0
2	GND
3	VCC + (8V – 15V)

Belegung SL3 Output	
Pin	Verwendung
1	Ausgang 1
2	Ausgang2
3	Ausgang 3
4	Ausgang 4
5	GND