Wire.h – Library für I2C

A.Schultze, DK4AQ, 19.04.2013

Wire.h - Library

- **Übersicht der Funktionen**
- **Zuordnung von Funktionsaufrufen zum Telegramm**
- **Detailbeschreibung**
  - **Wire.begin(adress)**
  - **Wire.requestFrom(address, count)**
  - **Wire.beginTransmission(address)**
  - **Wire.endTransmission()**
  - **Wire.send(value)**
  - **Wire.available()**
  - **Wire.receive()**
  - **Wire.onReceive(handler)**
  - **Wire.onRequest(handler)**

Wire.h – Library für I2C

Nach http://playground.arduino.cc/Main/WireLibraryDetailedReference

Die Wire-Library wird verwendet um die Hardware des ATmega328 zu initialisieren und einen einfachen Zugriff auf die Schnittstelle zu ermöglichen. Die Library arbeitet standardmässig mit 100kb/s.

Übersicht

Wire.begin(adress)	Initialisierung der Library
Wire.requestFrom(address, count)	Vorbereitung Empfang von einem Slave
Wire.beginTransmission(address)	Vorbereitung Senden, Schreiben Bytes in den Empfangspuffer akzeptieren
Wire.endTransmission()	Durchführung des Sendens aller Bytes aus dem Sende-Puffer und beenden Telegramm
Wire.send(value)	Schreiben eine Bytes in den Sende-puffer
Wire.available()	Liefert Anzahl der noch verfügbaren empfangenen Bytes

Wire.h – Library für I2C

Wire.receive()	Holt Bytes aus dem Epfangpuffer
Wire.onReceive(handler)	Interrupt: Slave hat Byte empfangen, kann mit **wire.receive()** abgeholt werden
Wire.onRequest(handler)	interrupt: Sende-Hardware fordert Reaktion des Programms an, meist Bedienung durch **wire.send()**

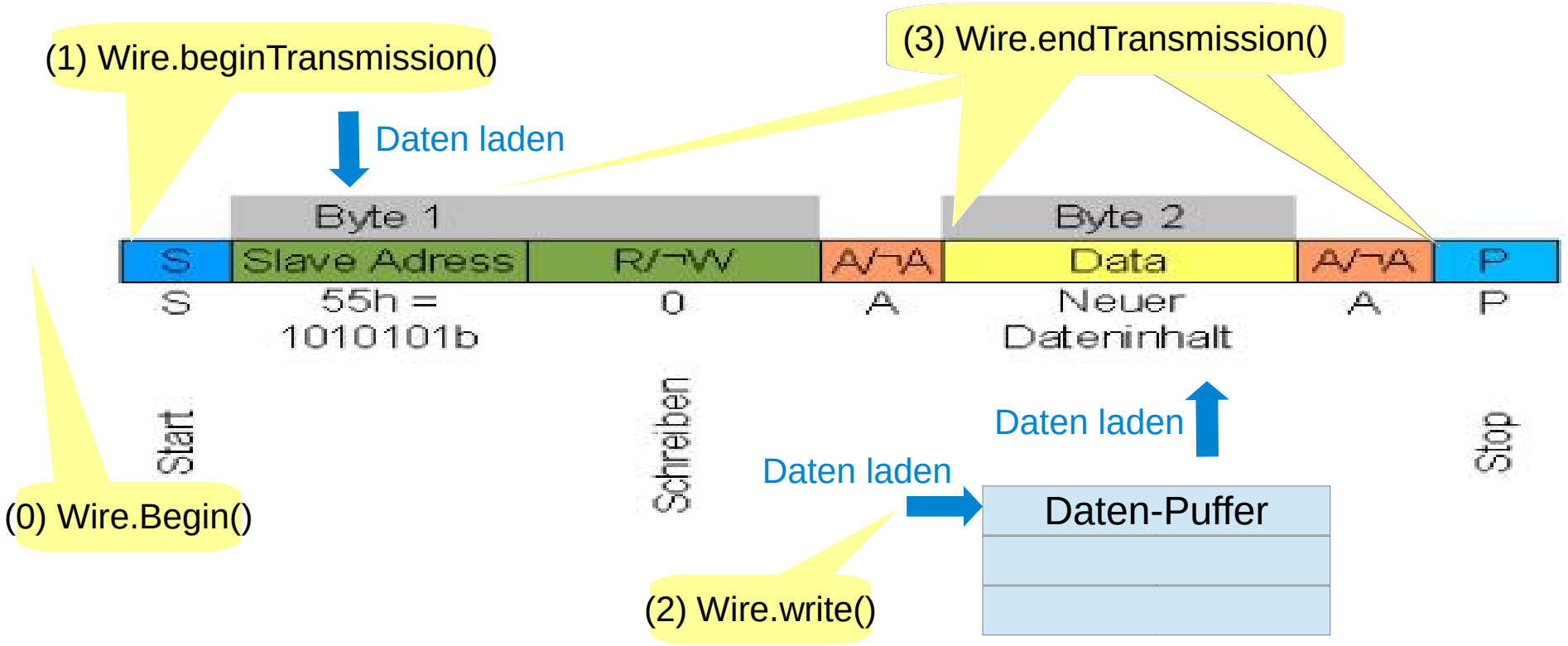
Wire.h:

http://code.google.com/p/arduino/source/browse/trunk/libraries/Wire/Wire.h?r=1092

Wire.ccp:

http://code.google.com/p/arduino/source/browse/trunk/libraries/Wire/Wire.cpp?r=1092

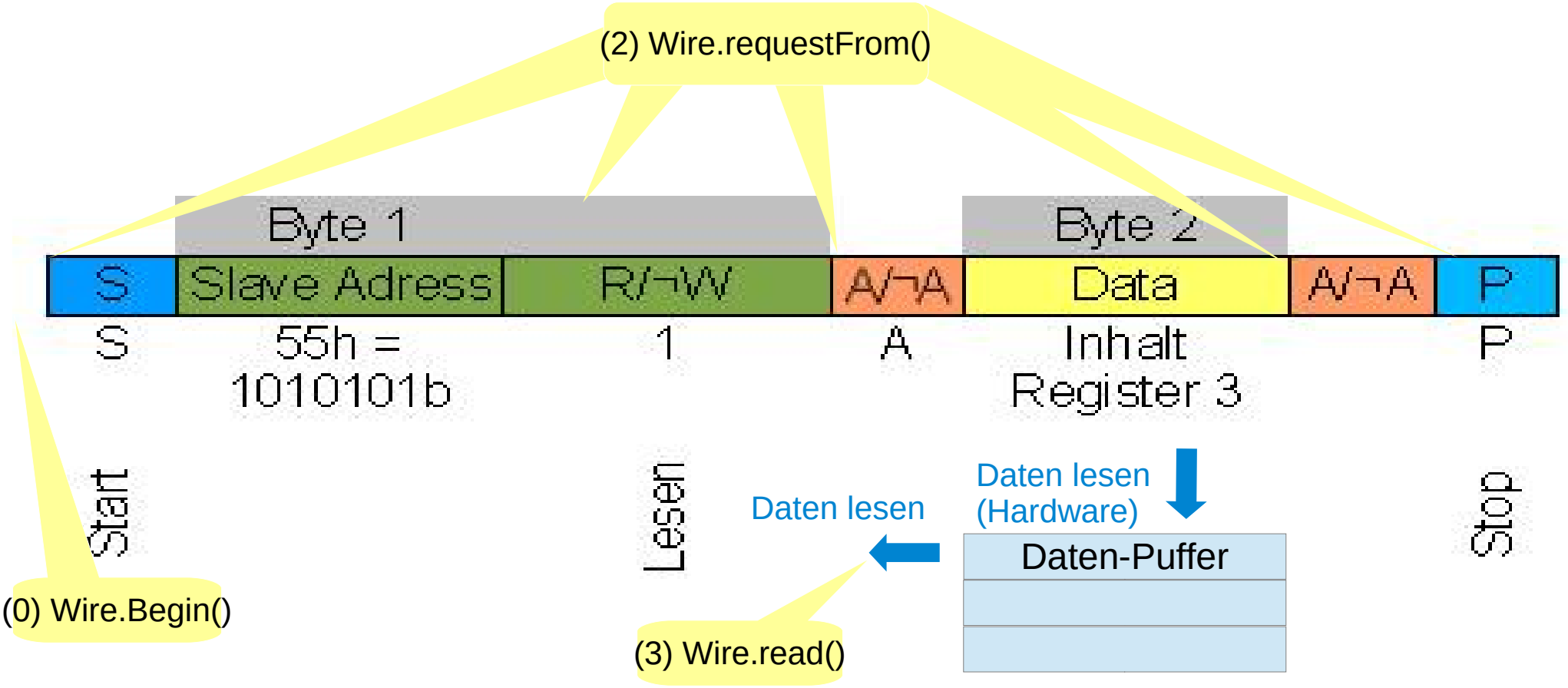
Wire.h – Library für I2C



Achnowledge/NotAcknowledge kommt vom Slave

Rolle der Wire-Funktionen beim Schreiben des Masters

Wire.h – Library für I2C



Acknowledge/NotAcknowledge kommt vom Master

Rolle der Wire-Funktionen beim Lesen des Masters

Wire.h – Library für I2C

Wire.begin(adress)

Master
Slave

- Setzt die Variablen der Wire-Library
- Setzt die Übertragungsgeschwindigkeit auf 100 kb/s für den Betrieb als Master (kann verstellt werden).
- Bei Betrieb als Slave wird mit **Wire.begin(adress)** die eigene Slave-Adresse angegeben
- Weist die richtigen Portpins zu (AT328 SDA: Pin4, SCL: Pin5 (hardwaremässig festgelegt)
- Enabled die Hardware

Wire.beginTransmission(address)

Master

- Fordert den Master zur Sendebereitschaft an den Slave **adress** auf.
- Setzt interne Variablen entsprechend
- Bereitet für die Funkltion **Wire.send()** vor
- Keine Hardware-Aktivitäten !

Wire.h – Library für I2C

Wire.requestFrom(address, count) Master

- Master fordert den Slave mit der Adresse **adress** dazu auf sendebereit zu sein.
- **adress** wird in einem Byte als 7-bit-Adresse gewertet, weitere Bits werden bei der Übergabe abgeschnitten. Der Wert wird danach um 1 bit nach links verschoben.
- **count** kündigt die Anzahl der anzufordernden Bytes an.
- **count** ist in Wire.h derzeit auf einen 32 byte Puffer begrenzt
- Es wird die **START condition** gesendet, dcer Master taktet mit SCL. Der Aufruf wartet, dass der Pufferspeicher wie angefordert gefüllt wird.
- Wenn der I2C-Empfänger die Übertragung der Bytes nach dem letzten Byte mit **NACK.** antwortet, wird **requestFrom()** sauber abgeschlossen.

Wire.h – Library für I2C

Wire.endTransmission()

Master

Die Bytes sollten durch **Wire.send()** vorher in den Sendepuffer geschrieben worden sein. Der Master beginnt (erst jetzt !) mit der Sendung auf den I2C-Bus und sendet solange bis alle Bytes des Puffers übertragen sind.

Es werden folgende Werte als Rückgabeparameter zurückgeliefert:

1. Erfolgreich gesendet
2. Sendpuffer ist zu groß (Probleme in Wire.h) Addresse wurde gesendet und NACK empfangen. Dies ist ein Fehler (z.B. falsche Adresse) und der Master soll eine STOP condition ausgeben
3. Datenbytes wurden gesendet und ein NACK empfangen. D.h. Der Slave hat nichts mehr zui senden. Der Master kann ein STOP senden oder einen wiederholten START
4. Andere Fehler

Wire.h – Library für I2C

Wire.send(value)

Master
Slave

- Fügt neue Byte-Inhalte in den Send-Puffer des Masters ein
- Keine Aktivitäten in der I2C-Hardware !
- Weitere Art der Parameterübergabe:
  - Wire.send(string), Übergabe von ASCII-Zeichen, Länge wird vorher übergeben in count.
  - Wire. send(data, quantity), Übergabe eines Arrays von Bytes (symbol.Anfangsadresse → Name des Arrays) mit Anzahl der Bytes (max 32 s.o.)

Wire.available()

- Liefert die Anzahl der Bytes im Empfangspuffer zurück
- Sofern der Wert > 0 ist, dürfen weiter Bytes mit Wire.receive() aus dem Empfamgs-Puffer gelesen werden.
- Wird gefüllt vom :
  requestFrom()
  onReceiveService()
- Keine Aktivtäten der I2C-Hardware !

Wire.h – Library für I2C

Wire.receive()

Master

- Holt das nächste Byte aus dem Empfangspuffer, solange noch eines existiert (siehe **Wire.available()**)
- Liefert den Datenwert des empfangenen Bytes zurück
- Keine Aktivität der I2C-Hardware !

Wire.h – Library für I2C

Wire.onReceive(handler)

Slave

- Verbindet den Empfangsinterrupt der I2C-Hardware mit einem Interruipt
- Die Interrupt-Service-Funktion **void handler(int byteCount)** wird hinter loop() definiert
- Die Funktion ist für den Slave-Empfangsbetrieb gedacht
- In dieser Routine kann mit **Wire.receive()** das gerade angekommene Byte eingelesen werden.
- ByteCount ist eine globale Variable und enthält die Anzahl der eingetroffenen Bytes
- Warnung: Es gab Forenbeiträge zu Fehlfunktionen, evtl. bereits gelöst

Wire.onRequest(handler)

Slave

- Verbindung zu dem Sendeinterrupt der I2C-Hardware
- Wird im Setup() durchgeführt.
- Die Funktion ist für den Slave-Sendemode bestimmt.
- Interrupt wird ausgeführt nach “Slave Address plus the Read bit")
- Der zugehörige Interrupt-Service-Rolutine **void handler(void)** kann dann mit **Wire.send()** geschehen
- Warnung: Es gab Forenbeiträge zu Fehlfunktionen, evtl. bereits gelöst

Wire.h – Library für I2C

Anhang: Tuning Arduino I2C auf 400kb/s (Methode 1)

Nach Aufsetzen der Wire-Library wird das Taktregister in der Hardware nachträglich verändert.

http://arduino.cc/forum/index.php?topic=15097.0

Arduino I2C auf fast Mode 400kb/s

```
#include <Wire.h>
void setup()
{
  // let Wire initialize the Two Wire Interface and interrupts
  Wire.begin();

  // change the clock rate (behind Wires' back)
  TWBR = ((CPU_FREQ / 400000L) - 16) / 2;
  // now we're cookin at 400 Khz... woo hoo!
}
```

Einsetzen: CPU_FREQ = 12000000L

Wire.h – Library für I2C

Anhang: Tuning Arduino I2C auf 400kb/s (Methode 2)

Tuning der Library

http://arduino.cc/forum/index.php?PHPSESSID=b3a3a1da6b02946cd4987067ac98eede&topic=16793.msg122226#msg122226

Autor: wayoda
Wuppertal/Germany
Offline Offline
God Member
*****
Karma: 0
Posts: 869

Re: Faster I2C

Hi,
the atmega-hardware can do 400 KHz, but you have to tweak the Wire-library in file **hardware/libraries/Wire/utility/twi.h.**

Near the top of the file you see :
Code:

```
#ifndef TWI_FREQ
#define TWI_FREQ 100000L
#endif
```

If you change that to
Code:

```
#ifndef TWI_FREQ
#define TWI_FREQ 400000L
#endif
```

The I²C bus should run at 400kHz. But you also have to the delete the files

hardware/libraries/Wire/Wire.o
hardware/libraries/Wire/utility/twi.o

because the library must be re-compiled before it uses the new speed.
(This is done automaticaly when you open the Arduino-IDE)

Eberhard