LVAs 384.996 & 384.174
'Mikrocomputer LU'
'Mikrocomputer für Informatiker_innen'

Abby

v1.0 - $\frac{3}{3}$

2024

Inhaltsverzeichnis

1 Allgemeines **4**
- 1.1 Laboraufbau . . . 5
- 1.2 Inbetriebname . . . 5
- 1.3 Logic Analyzer . . . 5
- 1.4 Funktionsweise des Abakus . . . 6

2 Hardware **7**
- 2.1 Lineareinheit . . . 7
- 2.2 Weitere Hardware . . . 8
- 2.3 Pinbelegung . . . 8
- 2.4 Schutzmaßnahmen . . . 9

3 Software **12**
- 3.1 SPI Konfiguration . . . 12
- 3.2 SPI Protokoll . . . 13
- 3.3 Schlittenbewegung . . . 14

4 Initialisierungsfahrt **15**

5 *Empfohlener* Übungsablauf **16**

6 Benotung **18**

A Rechenoperation **19**

Hinweis zur Laborübung: Bitte machen Sie sich *vor der Übung* mit folgenden Dokumenten vertraut:

- Diese Angabe (`Abby.pdf`).
- Die entsprechenden Abschnitte der im TISS hochgeladenen Unterlagen zum verwendeten NUCLEO-Board, konkret:
  - RCC
  - GPIO (inkl. der 'Alternate Functions')
  - SPI*
  - USART*
  - TIMER*
  - Interrupts (von Peripherieeinheiten)
  - EXTI* (inkl. SYSCFG)
  - ADC*

  *) Überlegen Sie, welche der gekennzeichneten Peripherieeinheiten (zB welcher Timer, welche USART) für die Anwendung in Frage kommen (das ergibt sich auch aus der Pinbelegung, siehe Tabelle 1).

Bei auftretenden Fragen während Ihrer Vorbereitungen wenden Sie sich *vor* Ihrem Übungstermin an die Tutoren.

1 Allgemeines

Ziel dieser Aufgabe ist es, den Umgang mit externer Peripherie zu erlernen. Insbesondere steht bei diesem Laboraufbau die Verwendung einer SPI-Schnittstelle in Kombination mit Timern und der simultane Umgang mehrerer ADC Kanäle im Vordergrund.

Abby ist ein elektromechanischer Abakus, bestehend aus sechs übereinander liegenden Kugelreihen mit je zehn Kugeln, die durch einen Mechanismus im Inneren der Maschine erkannt und verschoben werden können. Das Ziel der Aufgabe ist es, ein Programm zu entwickeln, welches die Kugeln so manipuliert, dass damit eine Addition druchgeführt werden kann. Die Operanden einer Rechenoperation sollen durch händisches Verschieben der Kugeln angegeben werden können. Anschließend soll die Maschine die gewünschte Rechenoperation durch automatisches Verschieben der Kugeln ausführen und am Ende das korrekte Ergebnis anzeigen.

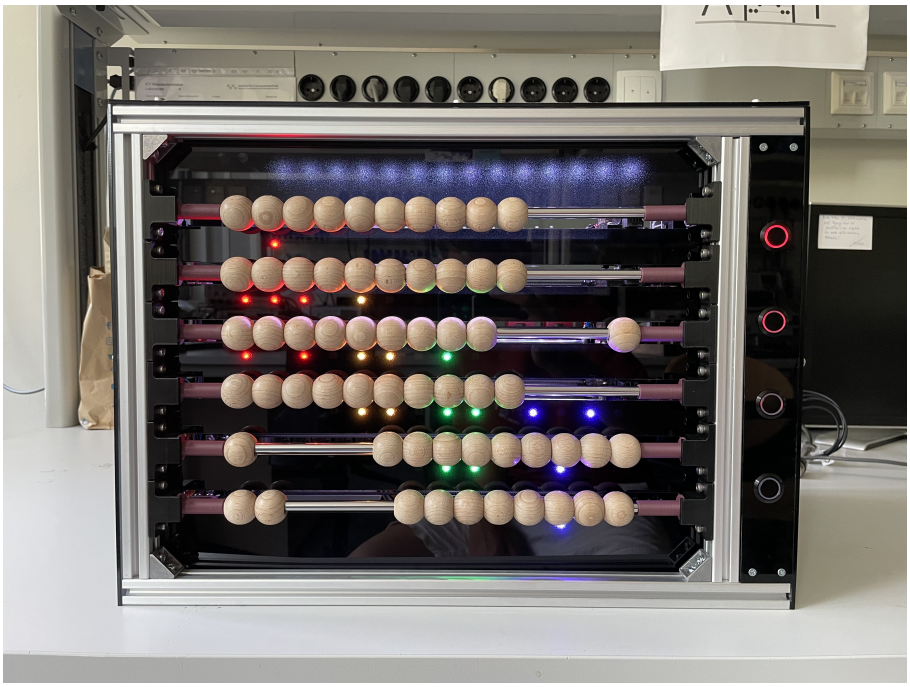

Abbildung 1: Laboraufbau Abby

1.1 Laboraufbau

Der Laboraufbau setzt sich aus folgenden Komponenten zusammen:

- Abby (in Abb. 1 dargestellt)
- STM32F334R8-Mikrocontrollerboard auf einer Adapterplatine

1.2 Inbetriebname

Stecken Sie zuerst die Stromversorgung für Abby an und verbinden Sie erst danach den Mikrocontroller mit dem Labor-PC über das USB-Kabel. Anderenfalls könnte es bei bereits aktivem Mikrocontroller zu einer Störung des Startvorgangs von Abby kommen. Treten während der Übung „unerklärbare“ Fehler auf, schalten Sie beide Komponenten aus, warten Sie einige Zeit bis sich alle Kondensatoren entladen haben und schalten Sie anschließend die Komponenten in der korrekten Reihenfolge wieder ein.

1.3 Logic Analyzer

Einige relevante digitale Pins sind an einem fix verbauten 8-Kanal Logic Analyzer angeschlossen. Mit dessen Hilfe kann das zeitliche Ein-/Ausgangsverhalten der Pins überprüft und zur Fehlerbehebung genutzt werden. Eine kurze Anleitung zum Umgang mit dem Logic Analyzer wird es zu Beginn des Labors geben, außerdem finden Sie eine ausfühliche Dokumentation in den hochgeladenen Unterlagen. Die Zuteilung der angeschlossenden Pins zu den acht Kanälen (LA CH0 .. CH7) ist in den Folgekapiteln sowie in Tab. 1 ersichtlich.

1.4 Funktionsweise des Abakus

Abby besteht aus sechs vertikal gestapelten Reihen mit je zehn Kugeln. Jeder Kugelreihe wird eine unterschiedliche Wertigkeit zugewiesen, die von unten nach oben aufsteigt. Zum Darstellen einer Zahl muss die entsprechende Anzahl an Kugeln an den rechten Rand der Stange geschoben werden. Soll zum Beispiel die Zahl 110 auf dem Abakus eingestellt werden, braucht es je eine Kugel in der Reihe mit Wertigkeit 10^2 und 10^1, sowie keine nach rechts geschobene Kugel mit Wertigkeit 10^0, wie in Abb. 2 dargestellt.

Der Abakus soll später auch für die Eingabe von zwei Operanden verwendet werden. In diesem Fall muss eine Trennreihe, gekennzeichnet durch eine Reihe mit zehn Kugeln, definiert werden (s. Abb. 8). Genaueres zum Ablauf einer Rechenoperation finden Sie im Anhang A.

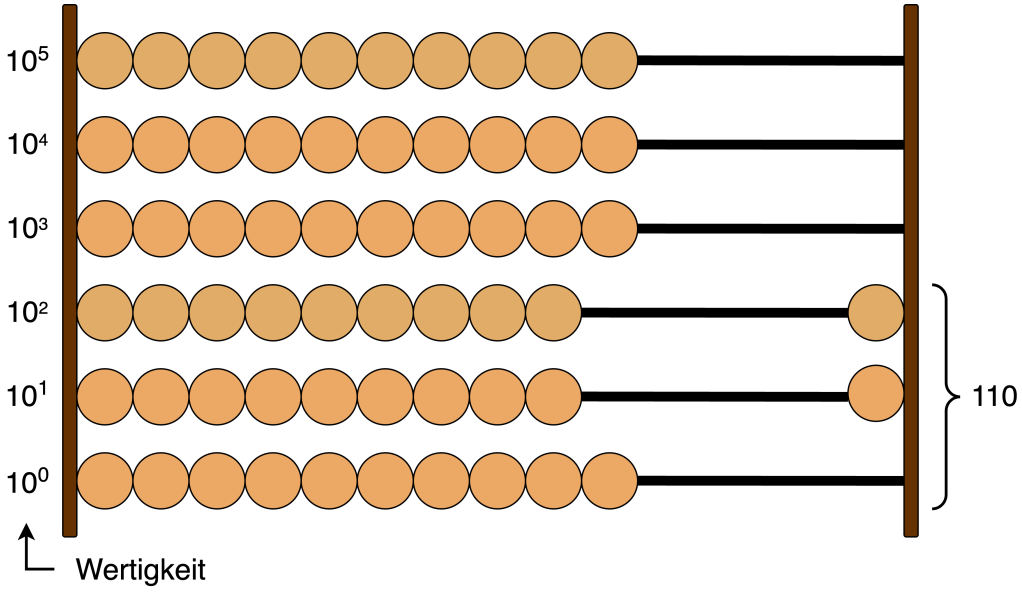

Abbildung 2: Darstellung der Zahl 110 auf einem Abakus

2 Hardware

2.1 Lineareinheit

Der Laboraufbau besteht aus sechs identisch aufgebauten Kugelreihen. Die schematische Zusammensetzung einer Reihe ist aus der Sicht von oben in Abb. 3 zu sehen. Unmittelbar hinter den Kugeln befindet sich ein Schlitten, der durch einen Schrittmotor parallel zur Kugelstange bewegt werden kann. Auf diesem Schlitten ist ein ein- und ausfahrbarer Stößel zum Bewegen der Kugeln sowie ein Infrarotsensor zum Detektieren der Kugeln montiert. Anders als in Abb. 3 dargestellt, befindet sich der Infrarotsensor im Laboraufbau direkt unter dem Stößel. Außerdem befindet sich am rechten Ende jeder Kugelreihe ein Endtaster, mit dem die „Home-Position“ des Schlittens bestimmt werden kann.

Mit dem Infrarotsensor ist es möglich, die Rückseite der Kugeln „abzutasten“, indem die Entfernung der Kugeloberfläche zum Sensor erfasst wird. Der Infrarotsensor gibt eine Spannung proportional zum reflektierten Infrarotlicht aus. Sie sollen diese Spannung messen und daraus die Positionen der zehn Kugeln bestimmen. Die Spannung beträgt etwa 0,2 V, wenn sich keine Kugel (also nur die Kugelstange) vor dem Sensor befindet und $\geq$2,6 V bei minimalem Abstand zwischen Kugelrückseite und Sensor. An einer Stelle, an der zwei Kugeln direkt aneinanderliegen, beträgt die Spannung etwa 0,6 V.

Wichtig: Fahren Sie den Stößel nur dort aus, wo Sie ein Minimum der Sensorspannung messen (0,6 V oder weniger), um eine Beschädigung des Stößels zu vermeiden!

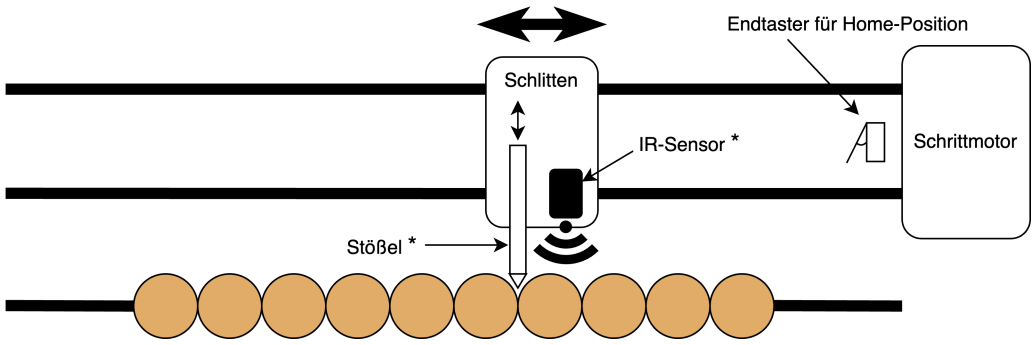

Abbildung 3: Aufbau einer Kugelreihe.
(*Stößel und IR-Sensor befinden sich direkt übereinander)

2.2 Weitere Hardware

Hinter dem schwarzen Acrylglas befindet sich für jede Kugelreihe eine LED-Leiste. Auf den LED-Leisten sind 14 RGB-LEDs angebracht, die sich unter jeder möglichen Kugelposition befinden. Zusätzlich gibt es zwei Leisten, die den Innenraum beleuchten können. Sie können die RGB-LEDs nach Ihrem Belieben verwenden, um diverse Lichteffekte zu generieren. Die Ansteuerung der LED-Pixel ist in Abschnitt 3.2 beschrieben.

Auf der rechten Seite der Kugelreihen befindet sich ein Bedienpanel mit vier beleuchteten Eingabetastern. Die Taster können Sie zum Beispiel dafür verwenden, um eine Rechenoperation zu starten.

2.3 Pinbelegung

Die Hardware ist über eine SPI-Schnittstelle und mehreren GPIO Ein- und Ausgängen mit dem Mikrocontroller verbunden. Die genaue Pinbelegung finden Sie in Tabelle 1. Die SPI-Pins müssen ihrer Verwendung entsprechend in den dafür vorgesehenen GPIO- und SPI-Registern konfiguriert werden.

Der Mikrocontroller ist über eine asynchrone Schnittstelle (UART) mit dem Labor PC verbunden. Die dafür verwendeten Pins sind ebenfalls in Tabelle 1 zu sehen. Die Konfiguration dieser Schnittstelle ist frei wählbar, solange Sie im Terminalprogramm und auf dem Mikrocontroller identisch konfiguriert

ist. Auf dem Labor-PC ist das Terminalprogramm „HTerm" vorinstalliert.

Der Pin PA0 dient als gemeinsames Schrittsignal für alle sechs Schrittmotoren. Dementsprechend muss dieser Pin so konfiguriert werden, dass auf diesem ein PWM-Signal generiert werden kann. Genaueres dazu können Sie im Abschnitt 3.3 nachlesen. Die zu SENS_0..5 zugehörigen Pins sind direkt mit den IR-Sensoren verbunden und müssen daher als analoge Eingänge konfiguriert werden. HOME_0..5 sind die **active-low** Verbindungen zu den Endtastern aller Kugelreihen. Diese sind als externe Interrupts zu konfigurieren. Die vier Eingabetaster rechts neben den Kugelreihen sind mit den Pins von INP_0..3 verbunden und sind dementsprechend als Eingänge zu konfigurieren. Es empfiehlt sich auch hier externe Interrupts zu verwenden. Die Eingabetaster verfügen über rote LED-Ringe, die über LED_0..3 angesteuert werden können.

Beachten Sie, dass die Pins für die Signale HOME_0..5 sowie INP_0..3 als Eingänge konfiguriert werden müssen!

Hinweis: Die Kugelreihen sind von 0 bis 5 nummeriert, wobei 0 die unterste und 5 die oberste Reihe bezeichnet. Die Eingabetasten sind von 0 bis 3 nummeriert, wobei 0 die oberste und 3 die unterste Taste bezeichnet.

2.4 Schutzmaßnahmen

Um Beschädigungen der Laborstation zu vermeiden, sind weitere Endtaster (zwei an der Kugelstange und einer am linken Ende der Linearführung) angebracht, auf die Sie keinen direkten Zugriff haben. Die Position aller Endtaster auf einer Lineareinheit können Sie aus der Abb. 4 entnehmen. Löst einer dieser Endtaster aus, so wird automatisch der Schlitten gestoppt und eine Kontrollleuchte auf der Vorderseite leuchtet auf. Diese Endtaster dienen nur zum Schutz des Aufbaus und sollen nicht von Ihnen verwendet werden, um die Aufgabe zu lösen (ausgenommen Initialisierungsfahrt s. Kap. 4).

STM32-Pin	Beschreibung	Funktionseinheit	LA CHx
PA2	TX	UART zum PC	
PA3	RX		
PA4	NSS	SPI	LA CH2
PA5	SCK		LA CH0
PA7	MOSI		LA CH1
PA0	STEP	Schrittsignal	LA CH3
PC0	SENS_0	IR-Sensor (analog)	
PC1	SENS_1		
PC2	SENS_2		
PC3	SENS_3		
PB0	SENS_4		
PB1	SENS_5		
PA1	HOME_0	Hometaster	
PB3	HOME_1		
PB4	HOME_2		
PB5	HOME_3		
PB6	HOME_4		
PB7	HOME_5		LA CH4
PB10	INP_0	Eingabetasten	
PB11	INP_1		
PB12	INP_2		
PB13	INP_3		
PC6	LED_0	LED-Ringe	LA CH6
PC7	LED_1		
PC8	LED_2		
PC9	LED_3		

Tabelle 1: Pinbelegung

Hinweis: Beim Erreichen des rechten Endes der Linearführung wird zwar der Schlitten (durch den Hometaster) automatisch gestoppt, Sie sollten aber dennoch den Stopp-Befehl an den Motor senden und mindestens 200 ms warten, bevor Sie den Schlitten erneut bewegen.

Hinweis: Der Hometaster prellt stark. Sie müssen in der Software eine entsprechende Entprellung vorsehen.

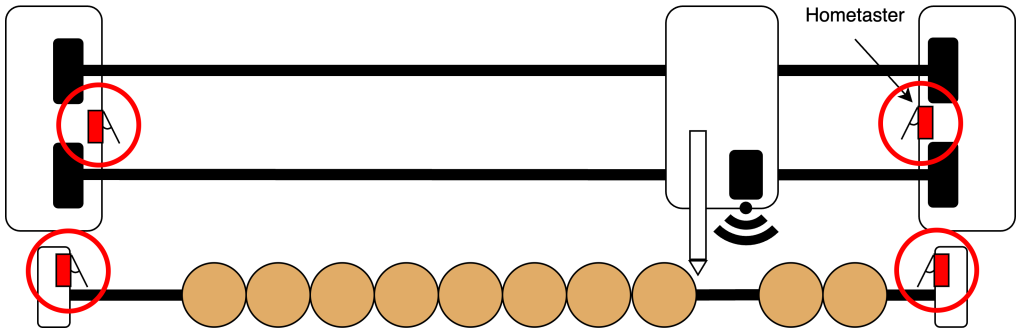

Abbildung 4: Position der Endtaster auf der Lineareinheit

3 Software

3.1 SPI Konfiguration

Ein Großteil der Kommunikation mit Abby erfolgt über die SPI-Schnittstelle, die in Abb. 5 zu sehen ist. Diese soll folgendermaßen konfiguriert werden:

- Modus: Master
- Richtung: Two-lines Full-Duplex
- Datengröße: 16 Bit
- Endian: MSB first
- Übertragungsrate: 500 kBit/s
- Idle clock line ist LOW
- Stabile Daten sollen bei steigender Flanke der Clock anliegen

Die NSS- (Slave Select-)Leitung muss vor der Übertragung jedes Datenwortes auf LOW und nach der Übertragung auf HIGH gezogen werden. Die Abb. 6 stellt die CLK, MOSI und NSS Signale zur Übertragung eines 16-Bit Befehls über die SPI dar.

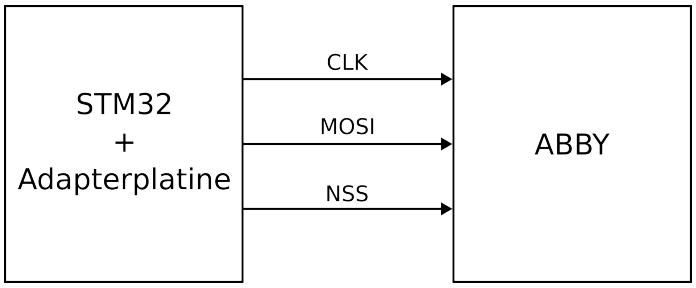

Abbildung 5: Schematischer Aufbau der SPI-Verbindung

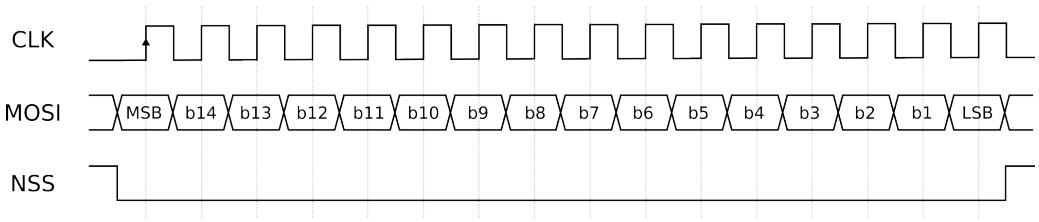

Abbildung 6: Übertragung eines 16-Bit SPI-Befehls

3.2 SPI Protokoll

In der Abb. 7 sind alle SPI Befehle von Abby aufgelistet. Alle Befehle haben eine Länge von 16 Bit. Es gibt drei Grundfunktionen, die durch die SPI-Befehle angesteuert werden können: die Bewegung der Schrittmotoren, das Ein- und Ausfahren der Stößel und das Ansteuern der RGB-LEDs. Die ersten zwei Bits jedes Befehls kennzeichnen die Operation (dunkelgrün). Alle Befehle beziehen sich jeweils auf eine Kugelreihe bzw. auf eine RGB-LED-Leiste. Auf die zwei Bits zum Kennzeichnen des Befehls folgen daher drei Bits, die die entsprechende Kugelreihe 0..5 beschreiben (blau). Die restlichen Bits b10..b0 beschreiben die Aktion, die ausgeführt werden soll. Bei dem Befehl „Schrittmotor bewegen“ ist das Bit b10 zuständig für das Fahren oder Stehenbleiben, b9 für die Richtung und b8 für das Microstepping. Für die Servomotoren gibt es nur ein Bit, das beschreibt, ob der Stößel ein- oder ausgefahren sein soll. Die Ansteuerung der RGB-LEDs geschieht über vier Bits (orange), die eine der RGB-LEDs auf der Kugelreihe auswählt und mit sechs darauffolgenden Bits, die jeden Farbanteil mit jeweils vier Helligkeitsstufen einstellen. Die LED Position 0 befindet sich auf der linken Seite jeder Kugelreihe.

	← MSB															LSB →
Aktion	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Schrittmotor bewegen	0	1	Kugelreihe 0-5			EN	DIR	Speed	X	X	X	X	X	X	X	X
Servo ausfahren	1	0	Kugelreihe 0-5			EN	X	X	X	X	X	X	X	X	X	X
LED ansteuern	1	1	Kugelreihe 0-5			LED Position 0-13				Rot		Grün		Blau		X
Alle LEDs einer Reihe ansteuern	1	1	Kugelreihe 0-5			1	1	1	1							X
Hintere Raumbeleuchtung	1	1	1	1	0	LED Position 0-13										X
Vordere Raumbeleuchtung	1	1	1	1	1	LED Position 0-13										X

Abbildung 7: SPI Befehle

3.3 Schlittenbewegung

Die Bewegung der Schlitten wird durch eine Kombination aus SPI-Befehlen und der Generierung eines PWM-Signals erreicht. Alle Schrittmotoren teilen sich ein gemeinsames STEP-Signal (s. Tab. 1). Bei jeder steigenden Flanke drehen sich die Motoren einen Schritt in eine gewünschte Richtung. Welcher Motor sich in welche Richtung dreht, wird durch Senden des Befehls „Schrittmotor bewegen“ (s. Abb. 7) bestimmt. EN=1 aktiviert den Schrittmotor, DIR=0 lässt den Schlitten nach links fahren und SPEED=0 wählt die normale Geschwindigkeitsstufe aus. Während ein Schlitten bereits in Bewegung ist, kann SPEED=1 ausgewählt werden, um kurzzeitig die Geschwindigkeit zu verdoppeln. Es empfiehlt sich zu Beginn immer SPEED=0 zu verwenden und als Bonusübung mit SPEED=1 zu experimentieren. Die Basisgeschwindigkeit aller Schrittmotoren wird durch die Frequenz des STEP Signals bestimmt, wobei die Frequenz maximal 10 kHz sein darf. Generieren Sie für die ersten Versuche ein 2,5 kHz PWM-Signal mit einer Pulsweite von 50%.

Zur Positionierung des Schlittens braucht es einen konstanten Referenzpunkt, den der rechte Endtaster darstellt. Berührt der Schlitten diesen Endtaster, soll ein Interrupt generiert werden, der den Schlitten sofort stoppt. Nun befindet sich der Schlitten in der sogenannten „Home-Position“. Durch Mitzählen der STEP-Flanken können Sie die exakte Position des Schlittens bestimmen. Welche Anzahl an Schritten welcher Distanz entspricht, wird durch die Initialisierungsfahrt bestimmt, s. Kap. 4.

4 Initialisierungsfahrt

Da die Länge der Lineareinheit nicht bekannt ist, müssen Sie am Anfang Ihres Programms eine Initialisierungsfahrt durchführen. Die Initialisierungsfahrt muss auf allen Kugelreihen durchgeführt werden und setzt sich aus folgenden Schritten zusammen:

1. Bringen Sie den Schlitten in die „Home-Position“ und fahren Sie anschließend den Stößel aus.

2. Bewegen Sie den Schlitten mit ausgefahrenem Stößel für eine ausreichend lange Zeit nach links, sodass alle Kugeln bis an den linken Rand geschoben werden. Dort wird Abby den Schlitten automatisch stoppen. Das ist das einzige Mal, dass ein anderer Endtaster als jener für die „Home-Position“ verwendet werden darf!

3. Fahren Sie den Stößel ein und bewegen Sie den Schlitten mit eingefahrenem Stößel so weit nach links, bis Sie das Ende der zehnten Kugel mit dem Sensor detektiert haben.

4. Bewegen Sie den Schlitten nun nach rechts und bestimmen Sie die Anzahl der Schritte, die der Schlitten bis zum Erreichen des Hometasters benötigt. Damit kennen Sie die Anzahl der nötigen Schritte für die gesamte Länge einer Kugelreihe. Diese Länge kann zwischen den Kugelreihen variieren, führen Sie daher die Initialisierungsfahrt individuell für jede Reihe durch.

5 *Empfohlener* Übungsablauf

Bauen Sie Ihr Programm modular auf! Implementieren und testen Sie die Teilaufgaben so weit wie möglich separat und führen Sie diese erst dann zur Gesamtlösung zusammen. Ein *Vorschlag* für die Herangehensweise und Separierung der Teilaufgaben:

1. Stellen Sie eine USART-Verbindung mit dem PC her. Über das vorinstallierte Terminal Programm können Sie so Befehle an den STM32 senden sowie Daten von diesem empfangen, um das Debugging Ihres Programms zu erleichtern. Konzentrieren Sie sich vorerst nur auf die Programmierung der obersten Kugelreihe (Nummer 5).

2. Stellen Sie die SPI-Verbindung zwischen dem STM32 und dem Laboraufbau her. Testen Sie die korrekte Funktion der SPI-Schnittstelle, indem Sie den Stößel des obersten Schlittens ein- und ausfahren lassen.

   Wichtig: Achten Sie beim Ausfahren des Stößels darauf, dass sich keine Kugel direkt vor dem Stößel befindet.

3. Generieren Sie ein PWM-Signal wie in Abschnitt 3.3 beschrieben. Verwenden Sie am Anfang eine Frequenz von 2,5 kHz. Je höher die Frequenz, desto schneller bewegt sich der Schlitten. Die Frequenz darf maximal 10 kHz betragen. Verwenden Sie die USART-Verbindung zum PC gemeinsam mit der SPI, sodass Sie über die PC-Tastatur den Schlitten und Stößel vorsichtig bewegen können.

4. Konfigurieren Sie einen externen Interrupt für den Hometaster und erstellen Sie eine Homing Routine wie in Abschnitt 3.3 beschreiben. Ihnen wird auffallen, dass der Taster seinen Zustand nicht sauber ändert, sondern prellt. Entprellen Sie den Taster in Ihrer Software.

   Wichtig: Warten Sie nach Erreichen der Home Position immer mindestens 200 ms, bevor Sie den Schlitten erneut bewegen.

5. Konfigurieren Sie den ADC und versuchen Sie mit dem Sensor Kugeln zu detektieren. Entwickeln Sie anschließend einen Algorithmus,

mit dem Sie feststellen können, ob sich der Sensor an einer Kugel vorbeibewegt. Mit dem entwickelten Algorithmus können Sie dann die Initialisierungsfahrt aus Kap. 4 implementieren.

6. Entwickeln Sie einen Algorithmus, mit dem Sie die Kugelpositionen einscannen können und anschließend die Kugeln so verschieben können, dass eine beliebige Zahl angezeigt werden kann. Es soll möglich sein die Kugeln mehrfach hintereinander zu verschieben, ohne dass die Kugeln erneut eingescannt werden müssen.

6 Benotung

Die Note für das Labor setzt sich aus dem Abgabegespräch sowie den Funktionalitäten, die Sie implementiert haben, zusammen. Im Folgenden befindet sich ein Richtwert, welche Funktionalitäten zum Erreichen einer bestimmten Note erfolgreich implementiert werden müssen. Dabei ist die Erfüllung aller Minimalanforderungen für die 'schlechteren' Noten die Voraussetzung für eine 'bessere' Note. Die Gesamtnote hängt jedoch zusätzlich von dem Abgabegespräch ab, d.h. wie gut Sie den Code erklären können und ob Sie in der Lage sind, kleine Änderungen vorzunehmen.

Die gesamte Steuerung soll über Interrupts erfolgen, wenn Sie glauben, an irgendeiner Stelle eine Warteschleife zu benötigen, fragen Sie bei den Tutoren nach, ob Sie das dürfen. Unnötige Warteschleifen führen zu Punkeabzügen.

Genügend:	Die Bewegung eines einzelnen Schlittens und das Ein- und Ausfahren des Stößels kann über Tastatureingaben am PC gesteuert werden. Außerdem kann der Schlitten zur Home Position zurückkehren.
Befriedigend:	Die Initialisierungsfahrt aus Kap. 4 wurde auf einem Schlitten implementiert. Der Laboraufbau kann eine Kugelanordnung als Zahl interpretieren und diese am PC ausgeben (via USART).
Gut:	Die vorherigen Punkte wurden so erweitert, dass sie auf allen Kugelreihen funktionieren. Alle Schlitten können gleichzeitig die Positionen der Kugeln bestimmen. Am PC kann eine mehrstellige Zahl eingegeben werden. Daraufhin sollen die Kugeln so verschoben werden, dass die eingegebene Zahl angezeigt wird.
Sehr gut:	Abby ist in der Lage auf Knopfdruck zwei beliebige (ein- oder mehrstellige) Zahlen, die direkt an den Kugelreihen eingestellt werden, zu addieren. Das Ergebnis der Addition wird zusätzlich mit den LEDs angezeigt. Zur Funktion einer Addition siehe Anhang A.

A Rechenoperation

Ihre Aufgabe besteht darin, eine Addition zu implementieren. Die beiden Summanden für die Addition sollen händisch auf den Kugelreihen eingestellt werden können. Beide Zahlen sollen mit einer Reihe von zehn rechts stehenden Kugeln voneinander getrennt werden. Zum Beispiel werden die Zahlen 11 und 99, wie in Abb. 8 zu sehen angegeben. Die Position der Trennreihe soll beliebig wählbar sein, sodass zum Beispiel auch eine vier- und einstellige Zahl miteinander addiert werden können.

Per Knopfdruck soll anschließend der Rechenweg bis hin zum Endergebnis möglichst visuell dargestellt werden. Als Beispiel sind die notwendigen Schritte für die Addition von 99 und 11 in Abb. 9 dargestellt. Im ersten Schritt werden die beiden Reihen mit Wertigkeit 10^0 zusammengezählt. Infolgedessen kommt es im zweiten und dritten Schritt zu einem Übertrag. Im letzten Schritt werden noch die Reihen mit Wertigkeit 10^1 addiert. Als Ergebnis sollte auf dem Abakus die 110 zu sehen sein wie in Abb. 2.

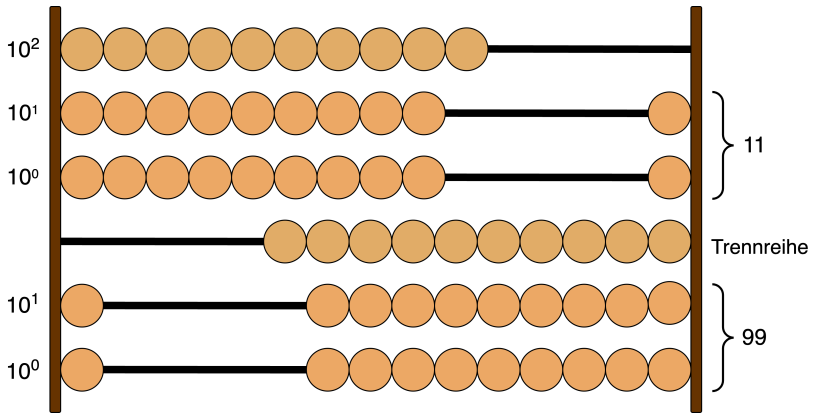

Abbildung 8: Manuelle Eingabe von 11 und 99 auf dem Abakus

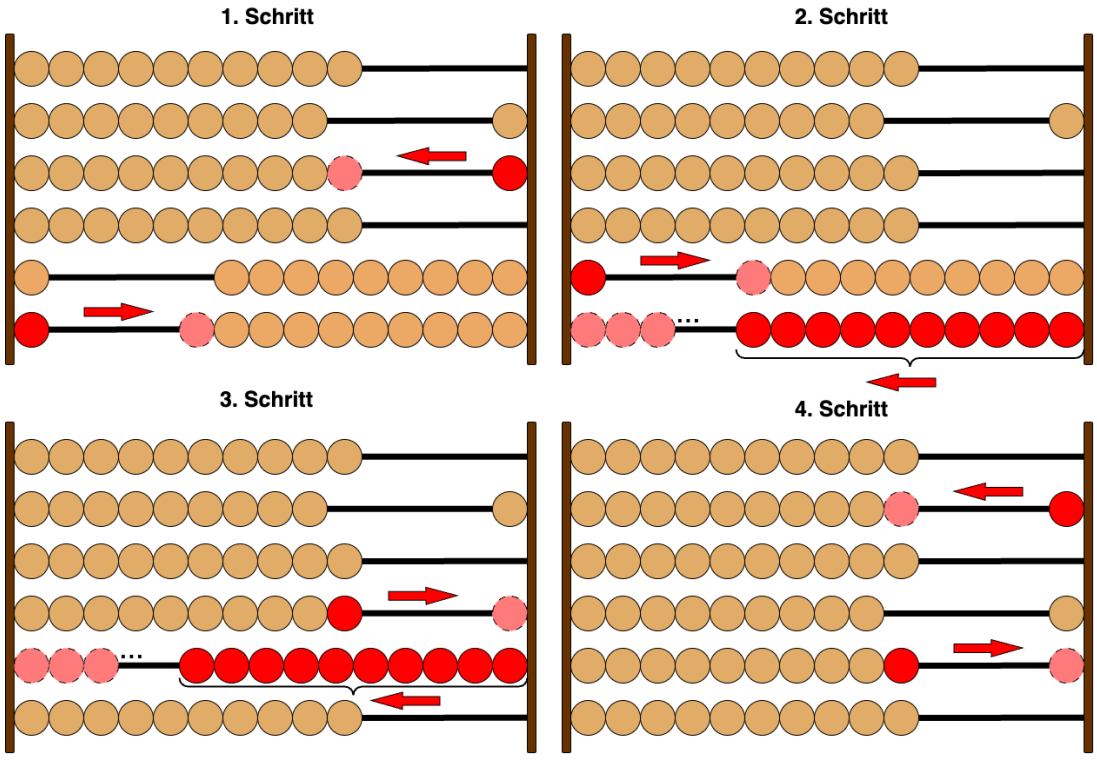

Abbildung 9: Schritte zur Addition von 11 und 99