Inhaltsverzeichnis

Abbildungsverzeichnis

Liste der Abkürzungen

HAL - Hardware-Abstraktionsschicht

LCD - Flüssigkristallanzeige

HCSR04 - Ultraschall-Abstandssensor (Modell)

I/O - Eingang/Ausgang

DDRD - Datenrichtungsregister - Port D

RS - Register Select (LCD-Steuerpin)

RW - Lesen/Schreiben (LCD-Steuerpin)

E - Enable (LCD-Steuerpin)

ISC - Ungeprüfter Interrupt

INT - Unterbrechung

TCNT - Timer/Zähler-Register

AVR - Erweitertes virtuelles RISC (Mikrocontroller-Architektur)

1 Einleitung

Im Bereich der Softwareentwicklung ist der Ruf nach vielseitigen Lösungen ungebrochen. Sich wiederholende Codierungsbemühungen behindern den Fortschritt und ersticken die Kreativität. Entwickler, die angesichts knapper Zeitpläne und begrenzter Budgets auf Effizienz abzielen, erkennen die Notwendigkeit von wiederverwendbarem und anpassungsfähigem Code. Die Hardware-Abstraktionsschicht (HAL) wird zu einem wichtigen Akteur bei der Bewältigung dieser Herausforderungen.

Der HAL dient als entscheidende Schnittstelle zwischen Anwendungscode und Mikrocontroller-Hardware und erleichtert die Wiederverwendung und Portabilität von Code über verschiedene Anwendungen und Plattformen hinweg. Dieser Bericht untersucht die praktische Implementierung eines HAL auf der Arduino-Plattform und demonstriert seine transformative Wirkung. Während wir uns durch die Feinheiten der Hardwareabstraktion navigieren, rationalisiert dieses Projekt den Entwicklungsprozess und bietet eine Grundlage für zukünftige Bemühungen, auf denen wir aufbauen können. Begleiten Sie uns bei dieser Erkundung der Hardwareabstraktion und ihrer wesentlichen Rolle in der Softwareentwicklung.

2 Ziele

1. **Grundlegendes zu HAL:** Machen Sie sich mit den grundlegenden Konzepten der Hardwareabstraktionsschicht (HAL) vertraut, um ihre Rolle bei der Verbesserung der Codeportabilität und Wiederverwendbarkeit über verschiedene Hardwarekonfigurationen hinweg zu verstehen.
2. **Grundlegendes zu Port-Manipulationen:** Gewinnen Sie Einblicke in Port-Manipulationen als Technik für den direkten Registerzugriff und erfahren Sie, wie sie Low-Level-Hardware-Interaktionen bei der Mikrocontroller-Programmierung vereinfachen.
3. **Entwicklung einer Hardware-Abstraktionsschicht:** Erstellen Sie eine funktionale und modulare HAL für die Mikrocontroller-Programmierung, die eine klare Abstraktion hardwarespezifischer Details für eine verbesserte Wartbarkeit und Vielseitigkeit des Codes gewährleistet.
4. **Verwendung von HAL zur Schnittstelle von Hardware:** Wenden Sie die entwickelte HAL an, um Hardwarekomponenten mit dem Mikrocontroller zu verbinden, um die Praktikabilität von HAL bei der Vereinfachung der Hardwareintegration und der Förderung der Codeanpassungsfähigkeit zu demonstrieren.

3 Grundlegendes zu HAL:

Ziel eins konzentriert sich auf die Entwicklung eines umfassenden Verständnisses der Hardware-Abstraktionsschicht (HAL). Dazu gehört es, die Prinzipien zu verstehen, die HAL zugrunde liegen, seine Bedeutung für die Förderung der Codeportabilität zu

erkennen und zu verstehen, wie es Low-Level-Hardware-Feinheiten abstrahiert, um eine optimierte und wiederverwendbare Codeentwicklung zu ermöglichen.

3.1 Grundlegendes zu Portmanipulationen:

Ziel zwei zielt darauf ab, Einblicke in Port-Manipulationen zu geben, eine wesentliche Technik für den direkten Registerzugriff in der Mikrocontroller-Programmierung. Die Untersuchung dieser Methode beleuchtet ihre Rolle bei der Vereinfachung von Low-Level-Hardwareinteraktionen, die es Entwicklern ermöglicht, Hardware-Peripheriegeräte direkt zu steuern und Code für bestimmte Mikrocontroller-Architekturen zu optimieren.

3.2 Entwickeln einer Hardware-Abstraktionsschicht:

Ziel drei ist die praktische Implementierung einer Hardware-Abstraktionsschicht (HAL) für die Mikrocontroller-Programmierung. Dieses Ziel konzentriert sich auf die Erstellung einer modularen und effizienten HAL, die hardwarespezifische Details abstrahiert und die Wartbarkeit und Anpassungsfähigkeit von Code über verschiedene Anwendungen und Plattformen hinweg erleichtert.

3.3 Verwenden von HAL zum Verbinden von Hardware:

1) Ziel vier betont die Anwendung des entwickelten HAL auf die Schnittstelle von Hardwarekomponenten mit dem Mikrocontroller. Durch die Demonstration der Praktikabilität von HAL bei der Vereinfachung der Hardwareintegration unterstreicht dieses Ziel die Vorteile der Verwendung von HAL zur Verbesserung der Codeanpassungsfähigkeit und der effizienten Hardwarenutzung in Mikrocontroller-Projekten.

4 HAL-Architektur

Die HAL-Architektur (Hardware Abstraction Layer) ist ein Entwurfsstil, der in der Softwareentwicklung eingesetzt wird, um die Lücke zwischen übergeordneten Softwarekomponenten und der zugrunde liegenden Hardware zu schließen. Die HAL fungiert als Zwischenschicht und bietet eine standardisierte Schnittstelle, die softwarespezifische Hardwaredetails der oberen Ebene abschirmt.

In der HAL-Architektur sind Low-Level-Hardwareoperationen und -konfigurationen innerhalb der HAL gekapselt, wodurch sichergestellt wird, dass der Rest der Software über eine Reihe klar definierter Funktionen oder Schnittstellen mit der Hardware interagiert.

Zu den Hauptzielen der HAL-Architektur gehören die Erleichterung der Codeportabilität über verschiedene Hardwarekonfigurationen hinweg, die Förderung der Wiederverwendbarkeit und die Verbesserung der Wartbarkeit durch die Isolierung hardwarespezifischer Details. Diese Abstraktionsschicht abstrahiert komplexe Hardwareinteraktionen, wie z. B. Operationen auf Registerebene, und bietet so eine einfachere und standardisierte Schnittstelle für Anwendungsentwickler.

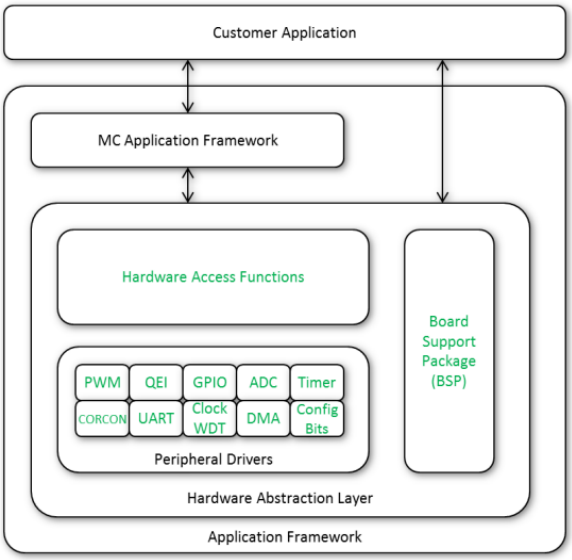

Abbildung 1 Blockschaltbild von HAL

[ref] http://ww1.microchip.com/downloads/en/DeviceDoc/hardware-abstraction-layer.pdf.

5 Arduino-Port-Architektur

Die Manipulation des Arduino-Ports beinhaltet die direkte Steuerung der I/O-Ports des Mikrocontrollers, um digitale Ein- und Ausgangsvorgänge effizient zu verwalten. Anstatt sich auf High-Level-Funktionen zu verlassen, ermöglichen Portmanipulationen eine detailliertere Steuerung, optimieren die Codeausführung und reduzieren den Ressourcenaufwand. Das Konzept dreht sich um die Manipulation von Port-Registern, die den Zustand einzelner Pins auf dem Mikrocontroller steuern.

Beim Aufbau einer Hardware-Abstraktionsschicht (HAL) unter Verwendung von Portmanipulationen liegt der Schwerpunkt auf der Schaffung einer modularen und unabhängigen Schicht, die übergeordneten Code von den Hardwarespezifika abschirmt. Die HAL kapselt Low-Level-Vorgänge, wie z. B. das Initialisieren von Ports, das Lesen und Schreiben auf Pins und das Konfigurieren von Pin-Modi. Durch die Abstraktion dieser Details wird die Hauptprogrammlogik portabler, da Änderungen an der zugrunde liegenden Hardware durch Anpassung der HAL berücksichtigt werden können, ohne den Rest des Codes zu beeinträchtigen.

In Arduino beinhalten Port-Manipulationen oft bitweise Operationen, um bestimmte Bits in Port-Registern zu setzen oder zu löschen. Wenn Sie beispielsweise einen digitalen Pin auf HIGH setzen, müssen Sie eine "1" in das entsprechende Bit im Portregister schreiben, während Sie ihn auf LOW setzen, indem Sie ihn auf "0" setzen. Diese direkte Manipulation von Registern verbessert die Effizienz und Reaktionsfähigkeit des Codes.

Die HAL, die mit Port-Manipulationen erstellt wurde, wird zu einer Brücke zwischen der Hardware des Mikrocontrollers und der übergeordneten Anwendung. Es erleichtert die Wiederverwendung von Code, erleichtert die Wartung und gewährleistet die Anpassungsfähigkeit über verschiedene Hardwarekonfigurationen hinweg. Durch den strategischen Einsatz von Port-Manipulationen erreicht der HAL ein Gleichgewicht zwischen Hardware-Steuerung und Abstraktion und bietet ein vielseitiges Framework für Mikrocontroller-basierte Projekte auf der Arduino-Plattform.

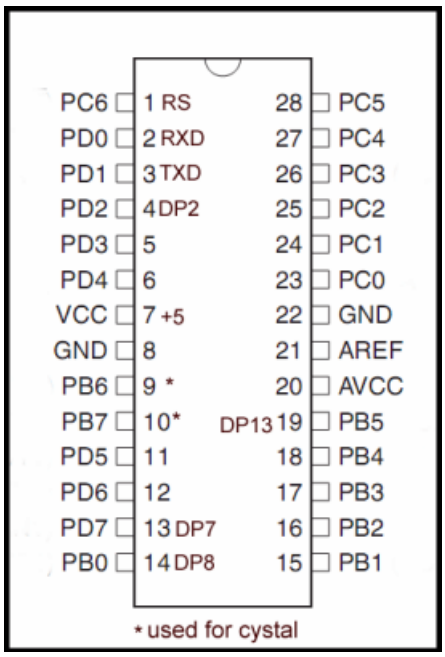

Abbildung 2 Arduino_328p Ports

[ref] https://www.bristolwatch.com/k150/port1.htm

6 LCD 16x2

Der 4-Bit-Modus eines LCD-16x2-Displays bezieht sich auf eine Kommunikationsmethode zwischen einem Mikrocontroller und dem LCD-Modul, bei der nur vier Datenleitungen (D4 bis D7) anstelle der herkömmlichen acht verwendet werden. Dieser Modus wird häufig verwendet, um Mikrocontroller-Pins zu schonen und die Verdrahtung in Projekten zu vereinfachen.

Der 4-Bit-Modus zum Lesen und Schreiben von Daten auf ein 16x2-LCD funktioniert wie folgt:

6.1 Initialisierung:

Um die Kommunikation zu starten, muss der Mikrocontroller das LCD-Modul initialisieren. Dabei wird eine Sequenz von Befehlen gesendet, in der Regel zunächst im 8-Bit-Modus, um die LCD-Anzeige zu konfigurieren. Im 4-Bit-Modus ist das LCD bereit für den Empfang von 4-Bit-Daten.

6.2 Schreiben von Daten auf die LCD-Anzeige:

Der Mikrocontroller sendet Daten an das LCD, indem er zwei separate 4-Bit-Nibbles sendet.

Jedes Nibble wird nacheinander auf die Datenleitungen (D4 bis D7) gesendet.

Wenn Sie z. B. den Binärwert 11011010 an die LCD-Anzeige senden möchten, senden Sie zuerst 1101 (D7-D4) und dann 1010 (D7-D4).

6.3 Steuersignale:

Zusätzlich zu den Datenleitungen verfügt das LCD-Modul über Steuerleitungen, darunter RS (Register Select), RW (Read/Write) und E (Enable).

- RS wählt aus, ob Daten oder eine Anweisung gesendet wird (0 für Anweisung, 1 für Daten).
- RW bestimmt die Richtung des Datenflusses (0 für Schreiben, 1 für Lesen).
- E initiiert den Lese-/Schreibvorgang von Daten, indem das LCD die eingehenden Daten einrasten lässt.

6.4 Schreibprozess:

- Setzen Sie RS auf 0 (Anweisungsmodus).
- Setzen Sie RW auf 0 (Schreibmodus).
- Senden Sie den High-Nibble der Daten.
- Pulsieren Sie die E-Linie, um das hohe Knabbern zu verriegeln.
- Senden Sie die Daten mit niedrigem Aufwand.
- Pulsieren Sie die E-Linie erneut, um das niedrige Knabbern zu verriegeln.

Der 4-Bit-Modus reduziert die Anzahl der erforderlichen Mikrocontroller-Pins und ist damit eine praktische Wahl für Projekte mit begrenzten Pins. Das Verständnis der Reihenfolge und des Timings der Datenübertragung ist entscheidend für eine erfolgreiche Kommunikation mit dem LCD im 4-Bit-Modus

7 Ultraschall-Sensor HCSR04

Wir konzentrieren uns darauf, wie Portmanipulationen genutzt werden können, um den HCSR04-Ultraschallsensor zum Laufen zu bringen, ohne in den spezifischen Code einzutauchen. Port-Manipulationen beinhalten die direkte Steuerung der I/O-Ports des Mikrocontrollers, was für die Verbindung mit Hardwarekomponenten wie dem HCSR04-Sensor von entscheidender Bedeutung ist.

7.1 Konfiguration des Trigger-Pins (trigPin):

Verwenden Sie Port-Manipulationen, um den Trigger-Pin als Ausgang festzulegen (z. B. DDRD |= (1 << trigPin); in AVR-Mikrocontrollern).

Manipulieren Sie das Portregister, um den Trigger-Pin auf HIGH oder LOW zu setzen, um den Ultraschallimpuls auszulösen

7.2 Konfiguration des Echo Pin (echoPin):

Verwenden Sie Port-Manipulationen, um den Echo-Pin als Eingang festzulegen (z. B. DDRD &= ~(1 << echoPin); in AVR-Mikrocontrollern).

Konfigurieren Sie externe Interrupts am Echo-Pin zur Erkennung logischer Änderungen während der Rückkehr des Ultraschallimpulses.

7.3 Timing-Messungen:

Verwenden Sie Timer-Module mit Port-Manipulationen, um die Zeitdauer zwischen dem Triggerimpuls und dem Rückecho zu messen.

Verwenden Sie den Zählerwert und die Umrechnungsfaktoren des Timers, um die Entfernung basierend auf der Schallgeschwindigkeit zu berechnen.

7.4 Interrupt-Behandlung für Echo-Pin:

Konfigurieren Sie externe Interrupts so, dass sie sowohl bei steigenden als auch bei fallenden Flanken des Echosignals ausgelöst werden.

Verwenden Sie Port-Manipulationen, um Interrupt-Flags zu behandeln und die Timing-Informationen zu verwalten.

7.5 Modulare Abstraktion:

Abstrahieren Sie die Sensorinteraktionen in Funktionen und kapseln Sie die Port-Manipulationen in dedizierte Funktionen.

Entwerfen Sie eine Hardwareabstraktionsschicht (Hardware Abstraction Layer, HAL), die Code auf höherer Ebene von Hardwaredetails auf niedriger Ebene abschirmt, wodurch der Code portabler wird.

Durch den Einsatz von Port-Manipulationen auf diese Weise kann der HCSR04-Sensor effektiv gesteuert werden. Die direkte Manipulation der Mikrocontroller-Ports ermöglicht eine effiziente und präzise Kommunikation mit dem Sensor, was genaue Abstandsmessungen ermöglicht. Dieser Ansatz erhöht die Modularität, macht den Code für verschiedene Mikrocontroller-Architekturen anpassbar und erleichtert die Integration in verschiedene Projekte.

8 Code-Struktur

Der Designprozess für die Aufteilung des Codes in vier Header-Dateien, .c- und .h-Dateien, speziell für LCD- und Ultraschallsensor-Funktionalitäten, folgt im Allgemeinen einem modularen und strukturierten Ansatz. Diese Trennung verbessert die Codeorganisation, Lesbarkeit und Wartbarkeit. Hier ist eine Aufschlüsselung des Designprozesses:

8.1 Hauptdatei (z. B. main.c):

8.1.1 Verantwortung

Orchestriert die Gesamtfunktionalität, initialisiert notwendige Peripheriegeräte und koordiniert die Interaktionen zwischen verschiedenen Modulen.

8.1.2 Design-Prozess:

- Include-Anweisungen: Fügen Sie die erforderlichen Header-Dateien für die LCD- und Ultraschallmodule hinzu.
- Funktionsaufrufe: Rufen Sie Funktionen von den LCD- und Ultraschallmodulen auf, um bestimmte Aufgaben auszuführen.
- Hauptfunktion: Enthält in der Regel die Hauptausführungsschleife und die Systeminitialisierung.

8.2 LCD-Modul (lcd.c und lcd.h):

8.2.1 Verantwortung

Verwaltet Interaktionen mit dem LCD-Display.

8.2.2 Design-Prozess:

- Header-Datei (lcd.h):
  Deklarieren Sie Funktionsprototypen für LCD-bezogene Operationen.
  Definieren Sie Konstanten oder Makros für LCD-Pins und -Konfigurationen.
- Quelldatei (lcd.c):
  Implementieren Sie Funktionen, die in lcd.h deklariert sind.
  Verwenden Sie Low-Level-Port-Manipulationen, um die LCD-Anzeige zu steuern.
  Stellen Sie Modularität und Unabhängigkeit von der Hauptprogrammlogik sicher.

8.3 Ultraschallmodul (ultrasonic.c und ultrasonic.h):

8.3.1 Verantwortung:

Verarbeitet Wechselwirkungen mit dem Ultraschallsensor.

8.3.2 Design-Prozess:

- Header-Datei (ultrasonic.h):
  Deklarieren Sie Funktionsprototypen für ultraschallbezogene Operationen.
  Definieren Sie Konstanten oder Makros für Ultraschallsensorkonfigurationen.
- Quelldatei (ultrasonic.c):
  Implementieren Sie Funktionen, die in ultrasonic.h deklariert sind.
  Verwenden Sie geeignete Techniken, um mit dem Ultraschallsensor zu kommunizieren.
  Stellen Sie Modularität und Unabhängigkeit von der Hauptprogrammlogik sicher.

9 Schlussfolgerung

Zusammenfassend lässt sich sagen, dass in diesem Projekt erfolgreich eine Hardware-Abstraktionsschicht (HAL) für die Arduino-Plattform implementiert wurde, die die Vielseitigkeit und Effizienz von Port-Manipulationen demonstriert. Durch die direkte Steuerung der Mikrocontroller-Ports erreichte das Projekt ein modulares Design, das die Organisation und Lesbarkeit des Codes verbesserte. Der HAL ermöglichte die Integration eines 16x2-LCD-Displays und eines HCSR04-Ultraschallsensors und demonstrierte die Praktikabilität der Abstraktion von Low-Level-Hardware-Interaktionen.

Der 4-Bit-Modus für das LCD-Display demonstrierte die Leistungsfähigkeit von Port-Manipulationen bei der Vereinfachung digitaler Ein- und Ausgabevorgänge. Dieser Modus, kombiniert mit einer gut strukturierten HAL, bot eine effektive Möglichkeit, eine Verbindung zum LCD herzustellen und gleichzeitig die Mikrocontroller-Pins zu schonen. Die Modularität des Codes gewährleistet die Anpassungsfähigkeit an verschiedene Projekte und Hardwarekonfigurationen.

Darüber hinaus veranschaulichte die Implementierung des HCSR04-Sensors die Präzision, die durch Portmanipulationen erreicht werden kann. Die direkte Steuerung von Trigger- und Echo-Pins in Verbindung mit Timer-Modulen ermöglichte genaue Entfernungsmessungen. Der HAL, der diese Low-Level-Details kapselte, betonte die

Abstraktion von Hardwareinteraktionen und förderte die Wiederverwendbarkeit und Wartbarkeit von Code.

Insgesamt liegt der Erfolg des Projekts in der Synergie zwischen Portmanipulationen und HAL-Designprinzipien. Das Ergebnis ist eine solide Grundlage für zukünftige Arduino-basierte Projekte, die die Bedeutung modularer Codestrukturen und effizienter Hardware-Abstraktion unterstreicht. Dieses Unterfangen zeigte nicht nur die praktische Anwendung theoretischer Konzepte, sondern beleuchtete auch das Potenzial, diese Prinzipien auf verschiedene Mikrocontroller-Projekte auszuweiten. Durch diese Untersuchung trägt der Bericht zum breiteren Diskurs über die Optimierung der Mikrocontroller-Programmierung bei, um die Flexibilität und Wartbarkeit von Code zu verbessern.

10 Bibliographie

M. Matijevic und V. Cvjetkovic, "Überblick über Architekturen mit Arduino-Boards als Bausteine für Datenerfassungs- und Steuerungssysteme", 2016 13th International Conference on Remote Engineering and Virtual Instrumentation (REV), Madrid, Spanien, 2016, S. 56-63. doi: 10.1109/REV.2016.7444440.

Popovici, K., Jerraya, A. (2009). Hardware-Abstraktionsschicht. In: Ecker, W., Müller, W., Dömer, R. (Hrsg.) Hardwareabhängige Software. Springer, Dordrecht. https://doi.org/10.1007/978-1-4020-9436-1_4

VMIL 2020: Proceedings of the 12th ACM SIGPLAN International Workshop on Virtual Machines and Intermediate Languages, November 2020, S. 5–14. DOI: 10.1145/3427765.3428495.

Beningo, J. (2017). Der Entwurfsprozess der Hardwareabstraktionsschicht. In: Entwicklung wiederverwendbarer Firmware. Apress, Berkeley, Kalifornien https://doi.org/10.1007/978-1-4842-3297-2_6

L. Loflin, "Port I/O Hardware and Software", Bristolwatch.com, 2024. [Online]. Verfügbar: https://www.bristolwatch.com/k150/port1.htm. [Zugriff: 24. Januar 2024]. Urheberrecht © 2024 Lewis Loflin.

Microchip Technology Inc., "MCU16", 2017. [Online]. Verfügbar: http://ww1.microchip.com/downloads/en/DeviceDoc/hardware-abstraction-layer.pdf. [Zugriff: 24. Januar 2024].

Netzwerk-Enzyklopädie, "Hardware Abstraction Layer (HAL)", zuletzt bearbeitet von der Redaktion, [online]. Verfügbar: https://networkencyclopedia.com/hardware-abstraction-layer-hal/. [Zugriff: 24. Januar 2024].