Lerne programmieren mit der Tello-EDU

Ein paar faszinierende Beispiele, um Programmieren zu lernen

Erstellt von

Lukas Seglias
Christian Nussbaum
Luca Berger
Luca Ritz

Fachhochschule
Bern

2020
PUBLISHER

Inhaltsverzeichnis

Kapitel 1

Einführung

1.1 Installation

Das folgende Setup wird vorausgesetzt:

- Windows 10
- IDE Visual Studio Community 2019 oder höher mit C++ [1]
- CMake in der Version 3.18.2 [2]
- Python in der Version 3 [3] (Füge Python der Pfad-Variable hinzu)

Als Alternative kann auch CLion [4] als IDE verwendet werden. Diese ist jedoch nicht frei erhältlich. Achtung, da auch CLion den MSVC++ verwendet, muss Visual Studio Community 2019 ebenfalls installiert werden.

1.1.1 MS Visual Studio 2019 konfigurieren

Sobald sichergestellt ist, dass obig aufgeführte Voraussetzungen gegeben sind, kann die IDE MS Visual Studio 2019 gestartet werden. Führe die folgenden Schritte durch:

- Auf der rechten Seite siehst du diverse Optionen, um ein Projekt zu öffnen oder zu erstellen. Wähle dort 'Lokalen Ordner öffnen' aus.
- Navigiere zu deinem Verzeichnis, in das du das Projekt gespeichert hast.
- Öffne den Projektornder (meistens 'learn_with_tello').
- Nun wird das Projekt geöffnet und die CMake-Generierung gestartet, welche eventuell fehlschlagen wird. Sollte dies der Fall sein, so ist die CMake-Version des Visual-Studios zu gering und wir müssen diese manuell einstellen. Öffne in dem Fall den Explorer und navigiere in deinen Projektordner. Dort siehst du die Datei 'CMakeSettings.json'. Öffne diese in einem Editor und füge die folgende Zeile hinzu:

[1] `https://visualstudio.microsoft.com/de/downloads/`
[2] `https://cmake.org/download/`
[3] `https://www.python.org/downloads/windows/`
[4] `https://www.jetbrains.com/de-de/clion/download/`

```
{
    ...
    "cmakeExecutable": "D:\\Prog\\CMake-3.18.2\\bin\\cmake.exe"
    ...
}
```

- Sobald diese Änderungen gemacht sind und du zurück ins Visual Studio wechselst, siehst du, dass CMake bereits die Generierung gestartet hat. CMake macht nun in einem ersten Schritt das Basissetup, um der IDE mitzuteilen, wie das Projekt aufgebaut und wie die 'targets' generiert werden sollen.

Sobald CMake fertig ist, kannst du nun die nachfolgend aufgeführten 'Startelemente' erstellen. Um dies zu bewerkstelligen, verschaffst du dir zuerst einen kleinen Überblick der IDE. Im Zentrum steht das Code-Fenster, welches den meisten Platz einnimmt. Auf der rechten Seite siehst du den Projektmappen-Explorer. Dort werden die verschiedenen Verzeichnisse sowie der Code gelistet. Oben kannst du einen grünen Play-Button sehen, daneben steht zu Beginn 'Startelement auswählen'. Klappe dieses auf und du solltest mindestens folgende Startelemente sehen:

00_base_module.dll Beinhaltet Basiseinstellungen für die Drohne(n)

01_keyboard_module.dll Die ersten paar Übungen.

01_keyboard_module_solution.dll Die Lösungen zu den Übungen.

99_template.dll Dies ist eine Vorlage für neue Module, das kann ignoriert werden.

app_basic.exe Die Hauptapplikation (Um diese zu starten, drücke den grünen Play-Button hinter dem Dropdown mit den 'targets')

app_common_video.dll Eine Bibliothek mit Code, den Module verwenden können.

Um ein Startelement zu kompilieren, muss dieses ausgewählt werden. Danach drückst du den Play-Button, um den Build zu starten.
Für den Anfang reicht es, wenn die folgenden Startelemente erstellt werden (Achtung, nach dem Build einer .dll wird ein Fehler angezeigt, dass diese nicht ausgeführt werden kann. Das ist ganz normal, klicke den Fehler einfach weg und schliesse die automatisch geöffnete Datei launch.vs.json ohne zu speichern):

- 00_base_module.dll
- 01_keyboard_module.dll
- app_common_video.dll
- app_basic.exe

Schliesse die nun geöffnete Anwendung.
Wenn die Builds allesamt ohne Fehler durchlaufen, dann kannst du deine Tello-EDU-Drohne starten. Du solltest sie zuerst mit der offiziellen App kalibrieren. Stelle sie danach auf den Boden, wo genug Platz drumherum und nach oben ist. Die Drohne sollte gelb blinken, das heisst,

sie wartet auf eine Verbindung. Wähle in den WLAN-Einstellungen das Tello-EDU WLAN aus. Wähle nun das Startelement 'app_basic' aus und drücke den Play-Button. Um deine Umgebung zu testen, drücke den Button "Take off" in der Applikation. Wenn die Drohne fliegt und du das Kamerabild der Drohne siehst, hast du alles richtig gemacht und kannst nun mit den Übungen starten. Drücke den Knopf nochmals, um die Drohne zu landen und schliesse die Applikation.

1.1.2 CLion konfigurieren

Wenn CLion als IDE verwendet wird, so müssen die folgenden Schritte durchgeführt werden. In einem ersten Schritt wird CLion geöffnet und über 'File - New Project' ein neues Projekt ersetllt. Nun muss das höchste CMakeLists.txt file des Repositorys 'learn_with_tello' ausgewählt werden. Danach kann die Build-Umgebung eingerichtet werden.

- Öffne 'File - Settings'
- Navigiere zu 'Build, Execution, Deployment'
- Wähle 'Toolchains' aus
- Prüfe, ob eine Toolchain namens 'Visual Studio' verfügbar ist. Wenn ja, erübrigen sich die weiteren Schritte.
- Drücke den 'Add'-Button in der oberen linken Ecke
- Nun muss die Umgebung ausgewählt werden. Beachte das nachfolgende Bild. Es muss mindestens eine CMake-Version 3.18.2 verfügbar sein.

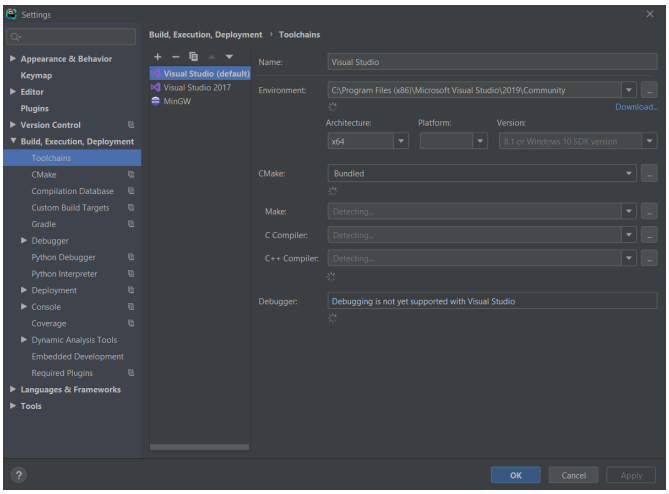

Abbildung 1.1: CLion toolchain Konfiguration

Nun bist du in der Lage, das Projekt im 'Debug'-Modus zu erstellen. Damit ist gemeint, dass zu jeder Library oder jeder ausführbaren-Datei (executable) zusätzlicher Code der Applikation hinzugefügt wird. Dadurch wird das Binary ungefähr zehn mal langsamer. Wenn du die Applikation im Release-Modus erstellen willst, dann musst du die folgenden Schritte durchführen. Im Release-Modus kann eine Applikation nicht mehr schrittweise durchlaufen werden, dafür ist sie viel performanter. Dieser Modus wird verwendet, wenn eine Applikation fertig ist und sich nicht mehr in der Entwicklungsphase befindet.

- Öffne 'File - Settings'
- Navigiere nach 'Build, Execution, Deployment'
- Wähle 'CMake' aus

- Prüfe, ob ein Profil namens 'Release' verfügbar ist. Wenn ja, dann ist bereits alles eingerichtet.
- Drücke den 'Add'-Button in der oberen linken Ecke.
- Ein neuer Eintrag wird erstellt. Benenne ihn 'Release', wenn dies nicht schon der Fall ist.

Nun werden die CMake-Konfigurationen neu erstellt. Nach diesem Vorgang, kannst du einen Blick in die obere rechte Ecke von CLion werfen. Dort siehst du ein grünes Hammersymbol. Daneben befindet sich ein Dropdown, wo die verschiedenen 'targets' (Ziele) ausgewählt werden können. Die Konfiguration wird dahinter ausgegeben. Dies ist entweder 'Debug' oder 'Release'. Wenn du das Dropdown ausklappst, kannst du sehen, dass du zwischen diesen Konfigurationen 'Debug' und 'Release' sowie allen 'targets' wechseln kannst. Ein 'target' wird in diesem Kontext als library oder executable-file verstanden. Es kann aber auch etwas spezielleres sein wie zum Beispiel ein Vorgang, um Dateien von einem Verzeichnis in ein anderes zu kopieren.
Die folgenden 'targets' sollten vorhanden sein (Es werden nicht alle aufgeführt):

00_base_module Beinhaltet Basiseinstellungen für die Drohne(n)

01_keyboard_module Die ersten paar Übungen.

01_keyboard_module_solution Die Lösungen zu den Übungen.

99_template Dies ist eine Vorlage für neue Module, das kann ignoriert werden.

app_basic Die Hauptapplikation (Um diese zu starten, drücke den grünen Play-Button hinter dem Dropdown mit den 'targets')

app_common Eine Bibliothek mit Code, den alle Module verwenden.

app_common_video Eine Bibliothek mit Code, den Module verwenden können.

Um ein 'target' zu kompilieren, muss dieses im Dropdown ausgewählt werden. Danach wird der grüne Hammer daneben geklickt. Für den Anfang reicht es, wenn die folgenden 'targets' erstellt werden:

- app_basic
- 00_base_module
- 01_keyboard_module

Wenn die Builds allesamt ohne Fehler durchlaufen, dann kannst du deine Tello-EDU-Drohne starten. Du solltest sie zuerst mit der offiziellen App kalibrieren. Stelle sie danach auf den Boden, wo genug Platz drumherum und nach oben ist. Die Drohne sollte gelb blinken, das heisst, sie wartet auf eine Verbindung. Wähle in den WLAN-Einstellungen das Tello-EDU WLAN aus. Wähle nun das 'target' 'app_basic' aus und drücke den Play-Button. Um deine Umgebung zu testen, drücke den Button 'Take off' in der Applikation. Wenn die Drohne fliegt und du ein Bild siehst, hast du alles richtig gemacht und kannst nun mit den Übungen starten. Drücke den Knopf nochmals, um die Drohne zu landen und schliesse die Applikation.

1.2 Architektur

Das Projekt besteht aus einer ausführbaren Applikation (app_basic), welche grundsätzlich nur alle Plugins lädt. Die Plugins bilden die eigentlich wichtigen Teile der Applikation, sie beinhalten die Übungen und Lösungen. Jedes Plugin ist eine in sich abgeschlossene Anwendung, welche für einen spezifischen Use-Case erstellt wurde, z.B. das Fliegen der Drohne mithilfe der Tastatur oder das Fliegen der Drohne über Objekterkennung.

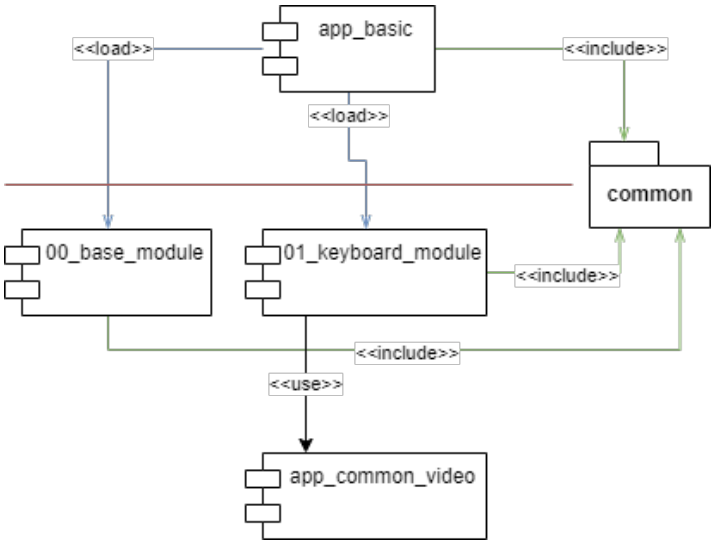

Abbildung 1.2: Architektur

Wenn du in einem der Module (Plugin) Änderungen gemacht hast, dann musst du dieses Modul neu kompilieren. Wähle dazu dieses Modul als Startlement in Visual Studio 2019 oder als 'target' in CLion aus und führe die Kompilierung durch. Wenn du anschliessend die Hauptapplikation startest, wirst du deine Änderungen sehen. Wenn du bei einer Aufgabe nicht weiter wissen solltest, kannst du einen Blick in die Referenzimplementation werfen. Diese enden mit der Bezeichnung '_solution'. Jeder Übungsabschnitt beinhaltet die entsprechenden Lösungen am Ende des Kapitels. Und nun, viel Spass bei den Übungen.

Kapitel 2

Grundverständnis

In diesem Kapitel befassen wir uns mit den ersten paar Aufgaben und wollen ein Grundlagenverständnis des Computers und seiner Funktionsweise aufbauen, welches wir später verwenden, um allfällige Probleme oder Verhaltensweisen besser verstehen zu können. Hierbei stellt sich also die Frage, wie so ein Gerät aufgebaut ist und wie die einzelnen Komponenten zusammenspielen, die wir hauptsächlich verwenden. Wir zeigen hier euch auch, was euch z.B. in einer Berufsschule erwarten könnte oder sogar erwarten wird. Ihr seht, womit ihr euch unter vielem anderen auseinandersetzen werdet.

Was du in diesem Kapitel sehen wirst:
☐ Was ist die CPU und wofür ist sie zuständig?
☐ Wie kann der Computer mit Zahlen arbeiten?
☐ Wie können zwei Zahlen addiert werden?
☐ Was ist eine logische Operation?
☐ Was ist die RAM und wofür ist sie zuständig?
☐ Wie kann ich mir etwas im Computer merken oder speichern?

2.1 Der Leader

Die CPU bildet das Herzstück eines Computers. Sie ist zuständig, um verschiedene Instruktionen auszuführen. Wenn z.B. zwei Zahlen addiert werden sollen, dann ist die CPU das Stück Hardware, welches diese Arbeit erledigt.

Abbildung 2.1: Intel CPU

Nun fragt man sich zurecht, wie die CPU denn solche Aufgaben bewältigen kann? Dazu muss man wissen, dass die CPU selbst wiederum aus mehreren Teilen besteht, die alle eine bestimmte Aufgabe erfüllen. Es sind dies im Wesentlichen:

Control unit Diese Einheit steuert den ganzen Ablauf der CPU. Sie sagt den anderen Komponenten im Gerät und in der CPU selbst, was sie wann tun sollen und steuert so den ganzen Computer.[Unb20d]

Arithmetic logic unit Wenn arithmetische oder logische Operationen durchgeführt werden sollen, dann ist diese Einheit wichtig. Wenn z.B. 4 + 5 gerechnet werden soll, dann wird die ALU damit beauftragt.[Unb20b]

Address generation unit Will die CPU auf den Hauptspeicher (RAM) zugreifen, dann wird in dieser Einheit die korrekte Adresse berechnet. Was das genau bedeutet wird im nächsten Abschnitt erklärt.[Unb20a]

Die CPU beinhaltet noch mehr solcher Einheiten, auf diese wird aber nun nicht mehr weiter eingegangen.

2.1.1 Übung 1 - Computer und Zahlen

Es wird Zeit für die erste kleine Übung. Wir wollen darin das Verständnis stärken, wie Zahlen überhaupt in einem Computer dargestellt werden und wie wir in der Lage sind, diese zu addieren.
Du hast sicher schon gehört, dass ein Computer mit 1en und 0en arbeitet. Wir wollen uns nun anschauen, was das genau bedeutet.
Dazu musst du dir aber erst mal darüber klar werden, wie wir Menschen uns Zahlen vorstellen, respektive, wie wie wir sie darstellen.
Wie ihr wohl sicherlich auch bereits gehört habt, arbeiten wir mit dem 10er-System. Das bedeutet nichts anderes, als dass wir zehn verschiedene Ziffern kennen, um eine Zahl darzustellen. In dem Fall 0 bis 9.
Nehmen wir nun als Beispiel die Zahl 145. Natürlicherweise interpretieren wir diese Zahl korrekt, wir machen uns aber nicht wirklich über den Aufbau an sich Gedanken. Wir wollen nun diese Zahl ein wenig anders formulieren und sie zuerst einmal auseinander nehmen. Dabei hilft uns das Wissen weiter, dass wir uns im 10er-System befinden (Basis 10).
Analysieren wir nun zuerst einmal unsere Zahl. Darin gibt es eine 1, diese steht ganz links an dritter Stelle. Sie bildet die Wertigkeit 100. Die zweite Ziffer ist eine 4, sie steht in der Mitte an

zweiter Stelle und bildet die Wertigkeit 10. Die dritte Ziffer steht ganz rechts an erster Stelle und bildet die Wertigkeit 1.

Ausgangslage dieser Zahl ist nun erstmal die Ziffer 1.
Die Frage stellt sich jetzt, wie wir die 1 an die Stelle ganz links verschieben können. Ganz konkret stellen wir uns die Frage, wie wir aus der Zahl 1 die Zahl 100 machen?
Wir haben dazu das Werkzeug der Multiplikation und können z.B. die 1 einfach mit der entsprechenden Wertigeit multiplizieren, wohin wir die Ziffer verschieben wollen. In dem Fall ist es die Wertigkeit 100:
$1 \times 100 = 100$

Serie 1:

1.1: Überlege dir kurz, wie wir die 4 an die Position in der Mitte verschieben können:
$4 \times \_\_\_\_\_ = 40$

1.2: Und nun überlege dir, was wir mit der 5 ganz rechts machen müssen?
$5 \times \_\_\_\_\_ = 5$

Sobald wir alle Ziffern korrekt verschoben haben, ergibt sich das folgende Bild:

$$\begin{aligned} 1 \times 100 &= 100 \\ 4 \times\ 10 &= \ 40 \\ 5 \times\ \ 1 &= \ \ 5 \end{aligned}$$

Wir stellen uns einmal mehr die Frage, wie wir nun aus diesen Einzelteilen unsere Zahl 145 erhalten?
Die Lösung ist in dem Fall in der Form der Addition gegeben. Sobald die Ziffern an die jeweils richtigen Positionen verschoben wurden, können wir diese einfach alle aufsummieren (addieren).

$$100 + 40 + 5 = 145$$

So weit, so gut. Wir merken uns:

- Zuerst verschieben wir die Ziffern alle mittels Multiplikation an die korrekte Position.
- Danach setzen wir die Zahl aus den einzelnen Resultaten mithilfe der Addition zusammen.

Nun wollen wir aber eigentlich nur noch mit der Basis 10 arbeiten. Das bedeutet, wir wollen bei der Multiplikation nicht die Wertigkeit an sich verwenden, sondern diese durch die Basis ausdrücken. Dazu müssen wir uns aber zuerst einmal in Erinnerung rufen, was ein Exponent ist. Diese kleinen Gleichungen sollte ein wenig Licht ins Dunkel bringen.
$m \times m = m^2$
$10 \times 10 = 10^2$
$2 \times 2 \times 2 = 2^3$

Mache dir dazu ein paar Gedanken und versuche die folgenden Übungen auszufüllen. Schreibe wenn möglich die Lösung mit Exponenten.

Serie 2:

2.1: $10 \times 10 =$_____ 2.3: $10 \times 5 =$_____

2.2: $10 \times 10 \times 10 =$_____ 2.4: $10^x = 1 \Rightarrow x =$_____

Da wir uns nun auch die Exponenten wieder in Erinnerung gerufen haben, können wir dieses Umhergeschiebe der Ziffern nun auch elegant mit der Basis ausdrücken.

$$\begin{aligned} 1\times10^2 &= 100 \\ 4\times10^1 &= 40 \\ 5\times10^0 &= 5 \end{aligned}$$

Das ganze können wir nun noch in Tabellenform überführen. In der ersten Zeile steht jeweils die Wertigkeit, in der zweiten Zeile der Faktor (die zu verschiebende Ziffer).

Wertigkeit	10^2	10^1	10^0
Faktor	1	4	5

Nun sind wir definitiv bereit, den nächsten Schritt zu tun. Wir wollen uns einige Gedanken zur Basis machen. Bis jetzt haben wir in unserem bekannten natürlichen Umfeld der Basis 10 gearbeitet. Der Computer hat aber eine kleine technische Einschränkung. Er arbeitet mit Strom. Er kennt also nur den Zustand »ich habe eine Ladung« und den Zustand »ich habe keine Ladung«. Wie wir richtig bemerkt haben, handelt es sich hierbei um ein binäres System, wir kennen also im Gegensatz zu unserem 10er-System nur zwei Zustände (0 und 1).[1]
Von diesem Moment an arbeiten wir also nur noch mit 0 und 1. Wir können die Ziffern 2 bis 9 nicht mehr verwenden, da diese gar nicht mehr existieren. Jetzt fragen wir uns natürlich zurecht, wie wir denn unsere 145 ausdrücken sollen ohne die 4 und die 5?
Dazu versuchen wir uns erst an einer anderen Zahl, nämlich einer Binären. Diese Zahl sieht nun folgendermassen aus:
10011
Nun denn, wir machen uns auch hier ein wenig mehr Gedanken über den Aufbau der Zahl. Wir erkennen, dass auch hier ganz links aussen eine 1 steht. Nun wissen wir bereits, wie wir im 10er-System eine Ziffer auf die richtige Position verschieben können. Glücklicherweise funktioniert dies hier genau gleich, es gibt nur einen Unterschied und zwar die Basis.

[1] Ein kleiner Exkurs: Die Darstellung der Ziffer in einem System spielt eigentlich keine Rolle. Wir könnten für die Zustände auch a und b verwenden. Wenn du dich dafür interessierst, dann schaue dir mal das »Martian number system« an. Suche entweder bei Google danach oder nütze folgenden link `https://mathematicscentre.com/taskcentre/098martn.htm`. Zu unseren Übungen gibt es auch noch eine gute Website, die vielleicht das eine oder andere noch ein wenig besser erklärt `https://www.freecodecamp.org/news/martian-math-812a029e2ea0/`

Serie 3:

Folgende Positionen sind innerhalb der Zahl gegeben:

Position	5	4	3	2	1
Ziffer	1	0	0	1	1

3.1: Verschiebe die Ziffer '1' an die Position 5 im 10er System.

_____

3.2: Verschiebe die Ziffer '1' an die Position 5 im 2er System.

_____

3.3: Verschiebe die Ziffer '0' an die Position 4 im 2er System.

_____

3.4: Verschiebe die Ziffer '0' an die Position 3 im 2er System.

_____

3.5: Verschiebe die Ziffer '1' an die Position 2 im 2er System.

_____

3.6: Verschiebe die Ziffer '1' an die Position 1 im 2er System.

_____

Wenn du mit den Übungen durch bist, hast du wahrscheinlich schon erkannt, dass du die Bestandteile der Zahl 10011 im binären System korrekt aufgebaut hast.

$$\begin{aligned} 1\times 2^4 &= 10000 \\ 0\times 2^3 &= \ 0000 \\ 0\times 2^2 &= \ \ 000 \\ 1\times 2^1 &= \ \ \ 10 \\ 1\times 2^0 &= \ \ \ \ 1 \end{aligned}$$

Wie im 10er-System auch erhalten wir hier die fertige Zahl über das Werkzeug »Addition«.

$$10000 + 0000 + 000 + 10 + 1 = 10011$$

Nun ist euch wahrscheinlich schon aufgefallen, dass die folgende Rechnung im 10er-System natürlich auch eine Lösung hat.

$$1 \times 2^4 = (10000)_2 \tag{2.1}$$

$$1 \times 2^4 = \ \ (16)_{10} \tag{2.2}$$

Die Zeile 2.1 beinhaltet das Resultat dargestellt im binären System mit 0 und 1. Siehe dazu die kleine 2 hinter der Klammer.
Die Zeile 2.2 beinhaltet das Resultat dargestellt im 10er-System mit den Ziffern 0 bis 9. Siehe dazu die kleine 10 hinter der Klammer. Und genau so funktioniert tatsächlich die Umrechnung von jedem beliebigen System ins 10er-System. Das bedeutet, dass die Zahl 10000 im binären der Zahl 16 im 10er-System entspricht.

Serie 4:

4.1: Ok, soweit so gut. Was ist jetzt aber mit der Zahl $(11)_2$ im Binären? Wie können wir so eine Zahl ins 10er-System umrechnen?
Mache dir dazu ein paar Gedanken und schreibe mal auf, wie du vorgehen würdest. Wenn du denkst, dass du entweder die Lösung gefunden hast oder du nicht mehr weiterweist, dann wirf auf jeden Fall einen Blick in die Lösungen. Dort wird der Vorgang nochmals ausführlich anhand eines Beispiels erklärt.

4.2: Rechne die folgende Zahl ins 10er-System um: $(100)_2$

4.3: Rechne die folgende Zahl ins 10er-System um: $(101)_2$

4.4: Rechne die folgende Zahl ins 10er-System um: $(10010001)_2$

4.5: Rechne die folgende Zahl ins 10er-System um: $(100)_3$

Wir haben nun also gesehen, wie wir Zahlen interpretieren und in anderen Systemen darstellen können. Weiterhin sind wir in der Lage von jedem beliebigen System aus ins 10er-System zu konvertieren.

2.1.2 Lösungen

Serie 1:

1.1: $4 \times 10 = 40$

1.2: $5 \times 1 = 5$

Serie 2:

2.1: $10 \times 10 = 10^2 = 100$

2.2: $10 \times 10 \times 10 = 10^3 = 1000$

2.3: $10 \times 5 = 50$

2.4: $10^0 = 1 \Leftrightarrow \{10^x = 1 \Rightarrow x = 0\}$

Serie 3:

3.1: $1 * 10^4 = 10000$

3.2: $1 * 2^4 = 10000$

3.3: $0 * 2^3 = 0000$

3.4: $0 * 2^2 = 000$

3.5: $1 * 2^1 = 10$

3.6: $1 * 2^0 = 1$

Serie 4:

4.1: Um die Zahl $(11)_2$ ins 10er-System zu überführen, betrachten wird die Zahl zuerst einmal wieder über ihre Bestandteile.
Wir nehmen sie also auseinander, wie wir das auch schon kennen.

$$1 \times 2^1 = (10)_2, (2)_{10}$$
$$1 \times 2^0 = (1)_2, (1)_{10}$$

Und wie wir auch schon wissen, können wir über die Addition die Zahlen wieder zusammenfügen. Dies tun wir nun ebenfalls:

$$(10)_2 + (1)_2 = (11)_2$$
$$(2)_{10} + (1)_{10} = (3)_{10}$$

Nun könnt ihr erkennen, dass die Lösung im binären System wieder die Ausgangszahl ergibt, wenn wir die Zahl aber im 10er-System interpretieren, dann ergibt dies uns 3, was genau der Zahl 11 im binären entspricht. Wir merken uns also:
Wir können eine Zahl in einer anderen Basis als 10 ganz einfach ins 10er-System überführen, wenn wir die Zahl in ihren Bestandteilen betrachten und die jeweilige Ziffer an dieser Position mit ihrer Wertigkeit multiplizieren. Haben wir das getan, so können wir über die Addition die Zahl wieder zusammensetzen.

4.2: $1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 = 4$

4.3: $1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 5$

4.4: $1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 145$

4.5: $1 \times 3^2 + 0 \times 3^1 + 0 \times 3^0 = 9$

2.1.3 Übung 2 - Einmal mehr eine Addition

Wir wollen nun noch einen Schritt weitergehen und uns einen Teil der Arbeit der ALU anschauen, nämlich die Addition zweier Zahlen im binären System. Um den Einstieg zu vereinfachen, wollen wir uns zuerst mit etwas auseinandersetzen, das wir bereits in der Unterstufe gelernt haben. Die schriftliche Addition im 10er-System.
Weisst du noch, wie das geht?

Serie 1:

1.1: Addiere die Zahlen 25 und 2 schriftlich. Wenn du nicht mehr weisst, wie das geht, dann schaue in den Lösungen.

1.2: Addiere die Zahlen 25 und 7.

1.3: Addiere die Zahlen 99 und 99.

1.4: Addiere die Zahlen 45 und 72.

Da wir jetzt ein wenig repetiert haben, können wir uns etwas Neuem widmen. Wir machen jetzt genau dasselbe, nur im binären System. Beachte dabei, dass wir wiederum nur die Ziffern 0 und 1 verwenden dürfen und der Überlauf auf die nächste Stelle bereits bei der Addition von zwei 1en passiert.

Serie 2:

2.1: Addiere die Zahlen 2 $(10)_2$ und 1 $(1)_2$ schriftlich im Binärsystem.

2.2: Addiere die Zahlen 3 $(11)_2$ und 1 $(1)_2$ schriftlich im Binärsystem.

2.3: Addiere die Zahlen 12 $(1100)_2$ und 22 $(10110)_2$ schriftlich im Binärsystem.

2.1.4 Lösungen

Serie 1:

1.1:
$$\begin{array}{r} 2\,5 \\ +\quad 2 \\ \hline 2\,7 \end{array}$$

1.2: Beachte den Übertrag von der ersten zur zweiten Stelle. 5 + 7 ergibt 12. Die zwei können wir an der Wertigkeit 1 belassen, wir übertragen aber eine 1 zur Wertigkeit 10.

$$\begin{array}{r} {\scriptstyle 1} \\ 2\,5 \\ +\quad 7 \\ \hline 3\,2 \end{array}$$

1.3:
$$\begin{array}{r} {\scriptstyle 1\,1} \\ 9\,9 \\ +\quad 9\,9 \\ \hline 1\,9\,8 \end{array}$$

1.4:
$$\begin{array}{r} {\scriptstyle 1} \\ 4\,5 \\ +\quad 7\,2 \\ \hline 1\,1\,7 \end{array}$$

Serie 2:

2.1:
$$\begin{array}{r} 10 \\ +\quad 01 \\ \hline 11 \end{array}$$

2.2:
$$\begin{array}{r} {\scriptstyle 1\,1\,0} \\ 11 \\ +\quad 01 \\ \hline 100 \end{array}$$

2.3:
$$\begin{array}{r} {\scriptstyle 1\,1\,1\,0\,0\,0} \\ 001100 \\ +\quad 010110 \\ \hline 100010 \end{array}$$

2.1.5 Übung 3 - Logische Operationen

Was uns jetzt noch fehlt ist das Verständnis von logischen Operationen. Es geht im Grunde nur darum, zwei Bedingungen auszuwerten und auf einen Wahrheitswert zu überführen. Nehmen wir als Beispiel einmal an, dass wir die folgenden Aussagen haben:

Aussage A Ein Flugzeug ist zum Tauchen gebaut worden.

Aussage B 9 ist durch 3 teilbar.

Aussage C 5 ist eine Primzahl.

Aussage D YB gewinnt in der nächsten Saison die Championsleague.

Wir haben offensichtlich zwei Aussagen (B und C), die der Wahrheit entsprechen und eine Aussage (A), die nicht zutrifft. Weiterhin gibt es eine Aussage (D), bei der wir den Wahrheitswert nur erahnen können, aber nicht genau kennen. Doch wir können festhalten, dass in der klassischen Aussagenlogik eine Aussage also entweder wahr oder falsch sein kann.[Unb20c]

Wir können diese Aussage nun verknüpfen und daraus eine Neue generieren. Dazu schauen wir uns zwei Werkzeuge an:

- AND (Und) $\wedge$
- OR (Oder) $\vee$

Angenommen, wir setzen die Aussagen B und C mithilfe des logischen AND zusammen, dann lautet die neue Aussage:
$B \wedge C$„9 ist durch 3 teilbar UND 5 ist eine Primzahl«
Diese Aussage ist ebenfalls wahr.

Serie 1:

Wir wollen nun mithilfe dieser Verknüpfungen ein paar Aussagen zusammensetzen. Kreuze das korrekte Resultat an.

1.1: $A \wedge B$
Wahr ☐ Falsch ☐

1.2: $A \vee B$
Wahr ☐ Falsch ☐

1.3: $B \vee C$
Wahr ☐ Falsch ☐

Zuletzt wollen wir uns noch kurz ansehen, wie wir diese Operationen auch auf binären Zahlen durchführen können und wie das Resultat aussieht. Dazu betrachten wir uns folgenden Zahlen, welche nun eine Aussage darstellen.

Aussage A $1101\ (13)_{10}$

Aussage B $1111\ (15)_{10}$

Um diese Operation nun korrekt durchführen zu können, brauchen wir eine Wahrheitstabelle, um nachschauen zu können, was bei bestimmten Kombinationen passiert.
Die Warheitstabelle für ein logisches AND sieht folgendermassen aus.

$\wedge$	0	1
0	0	0
1	0	1

Die Warheitstabelle für ein logisches OR sieht folgendermassen aus.

$\vee$	0	1
0	0	1
1	1	1

Wir führen nun ein logisches AND auf der Aussage A und B durch. Dies sieht folgendermassen aus:

$$\begin{array}{rl} & 1101 \\ \wedge & 1111 \\ \hline & 1101 \end{array}$$

Das Resultat dieser Operation ist also wieder $1101\ (13)_{10}$.

Serie 2:

Führe folgende logischen Operationen durch und bestimme auch gleich die Zahl im Zehnersystem.

2.1: $11001 \wedge 00110$

2.2: $11001 \vee 00110$

2.3: $1111 \wedge 0110$

2.1.6 Lösungen

Serie 1:

1.1: Wahr ☐ Falsch ☒

1.2: Wahr ☒ Falsch ☐

1.3: Wahr ☒ Falsch ☐

Serie 2:

2.1:
$$\begin{array}{rr} & 11001 \\ \wedge & 110 \\ \hline & 0 \end{array}$$
Dies entspricht $(0)_{10}$.

2.2:
$$\begin{array}{rr} & 11001 \\ \wedge & 110 \\ \hline & 11111 \end{array}$$
Dies entspricht $(31)_{10}$.

2.3:
$$\begin{array}{rr} & 1111 \\ \wedge & 110 \\ \hline & 110 \end{array}$$
Dies entspricht $(6)_{10}$.

Abbildungsverzeichnis

Literatur

[Unb20a] Unbekannt. *Address Generation Unit*. 2020. URL: https://en.wikipedia.org/wiki/Central_processing_unit#Address_generation_unit (besucht am 05. 10. 2020).

[Unb20b] Unbekannt. *Arithmetic Logic Unit*. 2020. URL: https://en.wikipedia.org/wiki/Central_processing_unit#Arithmetic_logic_unit (besucht am 05. 10. 2020).

[Unb20c] Unbekannt. *Aussagenlogik*. 2020. URL: https://de.wikipedia.org/wiki/Aussagenlogik (besucht am 07. 10. 2020).

[Unb20d] Unbekannt. *Control Unit*. 2020. URL: https://en.wikipedia.org/wiki/Central_processing_unit#Control_unit (besucht am 05. 10. 2020).

Glossar

IDE Eine integrierte Entwicklungsumgebung stellt eine Sammlung verschiedener Werkzeuge zur Softwareentwicklung zur Verfügung..

CMake CMake generiert Makefiles, welche Anweisungen zum Erstellen einer Software enthalten..

Python Python ist eine Programmiersprache.

Plugin Ein Plugin ist eine Softwarekomponente, die eine bestehende Software um Funktionalität erweitert. Diese wird zur Laufzeit eingebunden und kann nicht ohne die Hauptanwendung ausgeführt werden..

CPU Central Processing Unit.

ALU Arithmetic logic unit.