IMS
Institut für Mikroelektronische Systeme
Leibniz Universität Hannover

Leibniz Universität Hannover

Visual Studio Code

Programmierpraktikum TI

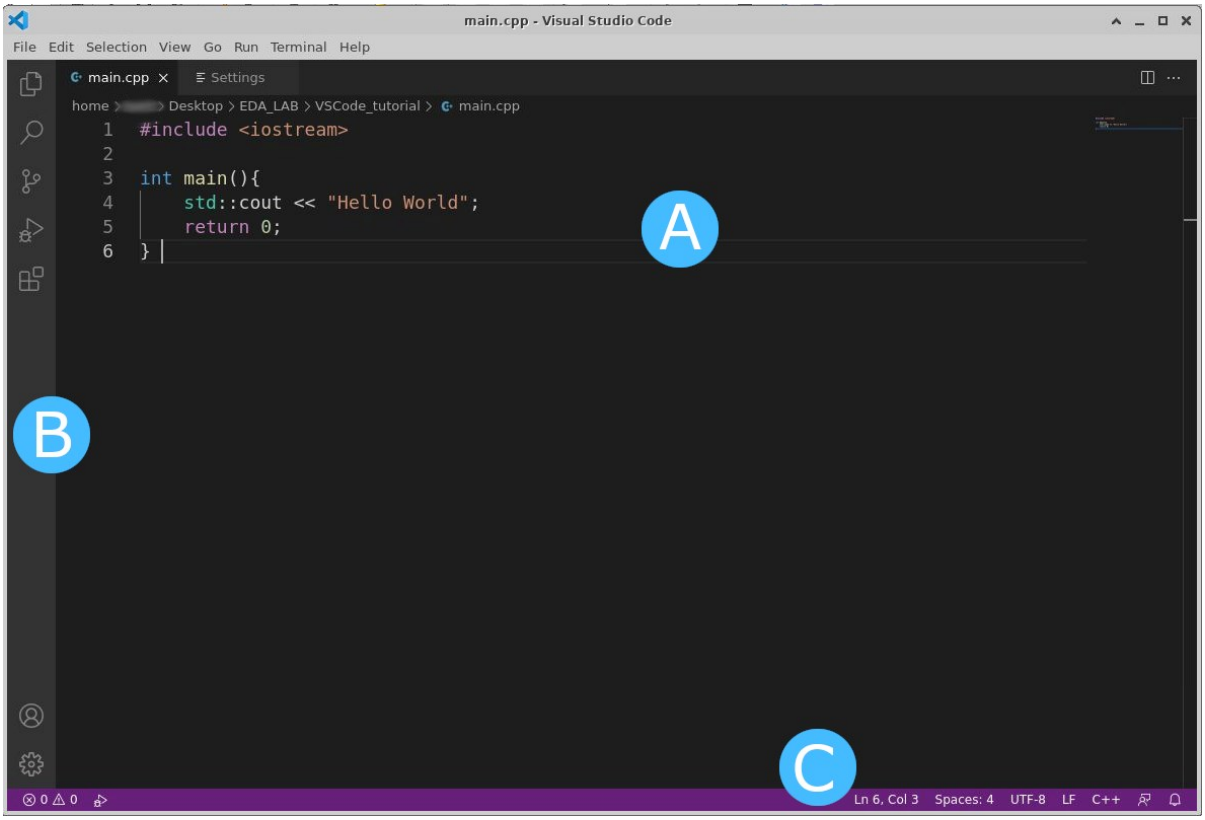
1. Einleitung

Im Rahmen des Programmierpraktikums und des Programmierprojekts Electronic Design Automation wird von den Studierenden verlangt, in C++ Aufgaben zu lösen. Hinsichtlich des näheren Bezugs zu heutigen Arbeitsumgebungen wurde dabei entschieden, von ursprünglich QTCreator auf VSCode umzusteigen. Auch wenn es den Studierenden mehr oder weniger freigestellt ist, welchen Editor sie nutzen, so macht die überschaubare Oberfläche, die Möglichkeiten, vieles anzupassen, und zusätzliche Tools in Form von Extensions zu installieren sowie die direkte Integration einer Konsole und Anbindung an Version Control (z.B. Git) VSCode zu einem einfach handhabbaren jedoch mächtigen Tool zur Programmierung.

In dieser Kurzanleitung soll ein genereller Überblick in die integrierte Entwicklungsumgebung Visual Studio Code (im Folgenden VSCode) gegeben werden. Eine ausführlichere Anleitung können Sie unter https://code.visualstudio.com/docs finden.

2. Die Oberfläche

Im Wesentlichem enthält die Oberfläche von VSCode drei wichtige Bereiche. Die hier beschriebene Version von VSCode kann allerdings geringfügig von der Ihren abweichen.



VSCode GUI

- **A: Editor Bereich**: Hier sind alle geöffneten Dateien zu finden. Sind mehrere geöffnet, werden diese ähnlich zu einem Webbrowser in Tabs aneinander angereiht und sind aufrufbar. Da immer nur eine Datei voll angezeigt werden kann, ist es möglich, den Editor in Teile zu splitten. Dies ist entweder über das Symbol oben rechts oder über ein mit Linksklick im Editorbereich auf einen Tab aufrufbares Dropdownmenu (z.B. “Split left”) möglich. Am rechten Rand wird zudem eine Minimap der geöffneten Datei angezeigt. Diese gibt mehr Überblick bei großen Dateien und erleichtert das Finden von im Code markierten Stellen. Hier geschriebener Code wird, nach erstmaligem Abspeichern mit der korrekten Endung, syntaktisch markiert sowie korrekt eingerückt. Ein Codecompletion-Vorschlag erscheint beim Schreiben.
- **B: Activity Bar**: Die auf der linken Seite positionierte Activity Bar wird im nächsten Kapitel genauer betrachtet. Im Groben lassen sich dort Bereiche aufklappen wie den File Explorer, die Suchfunktion, Version Control, Ausführ- und Debuggingtools sowie Extensions installieren. Zudem können über das Zahnrad weitere Einstellungen vorgenommen werden.
- **C: Status / Side Bar**: Die untere Bar dient dazu, Statusinformationen zu erhalten. Dort werden Fehler und Errors angezeigt, aber auch die Position des Cursors, ausgewähltes Encoding, Sprache und ähnliches.

Neben diesen Bereichen gibt es zudem ein Ribbon-Menü am oberen Rand, über das sich Funktionen zu File, Edit und anderen klassischen Menüpunkten aufrufen lassen. Über den dort befindlichen Menüpunkt “Terminal”, kann im unteren Editor-Bereich eine Konsole geöffnet werden. Diese ist standardmäßig bash und befindet sich direkt im aktuellen Projektordner.

3. Die Activity Bar

Explorer

 Der Fileexplorer dient dazu, einzelne Dateien oder ganze Ordnerstrukturen in VSCode zu laden. Es besteht die Möglichkeit, dies auch per Drag-and-Drop zu laden und auch ein Repository zu klonen. Die dort geladenen Strukturen sind wie ein typischer Filebrowser aufgebaut (Ordner auf/einklappbar etc.).

Search

 Über den Menüpunkt „Search“ können die Dateien durchsucht werden. So können dort auch Regular Expressions eingegeben werden oder auch bestimmt werden, welche Dateien durchsucht und welche von der Suche ausgeschlossen werden sollen.

Source Control

 VSCode bietet die Integration von Version Control. Dafür kann ein bereits geklontes Projekt geöffnet werden bzw. direkt der clone-Prozess durchgeführt werden. Ist ein mit Git verknüpfter Ordner geöffnet, können die Git-Befehle von VSCode aus ausgeführt werden. Vor allem können dort einfache Commits getätigt, sowie Changes eingesehen werden.

Run and Debug

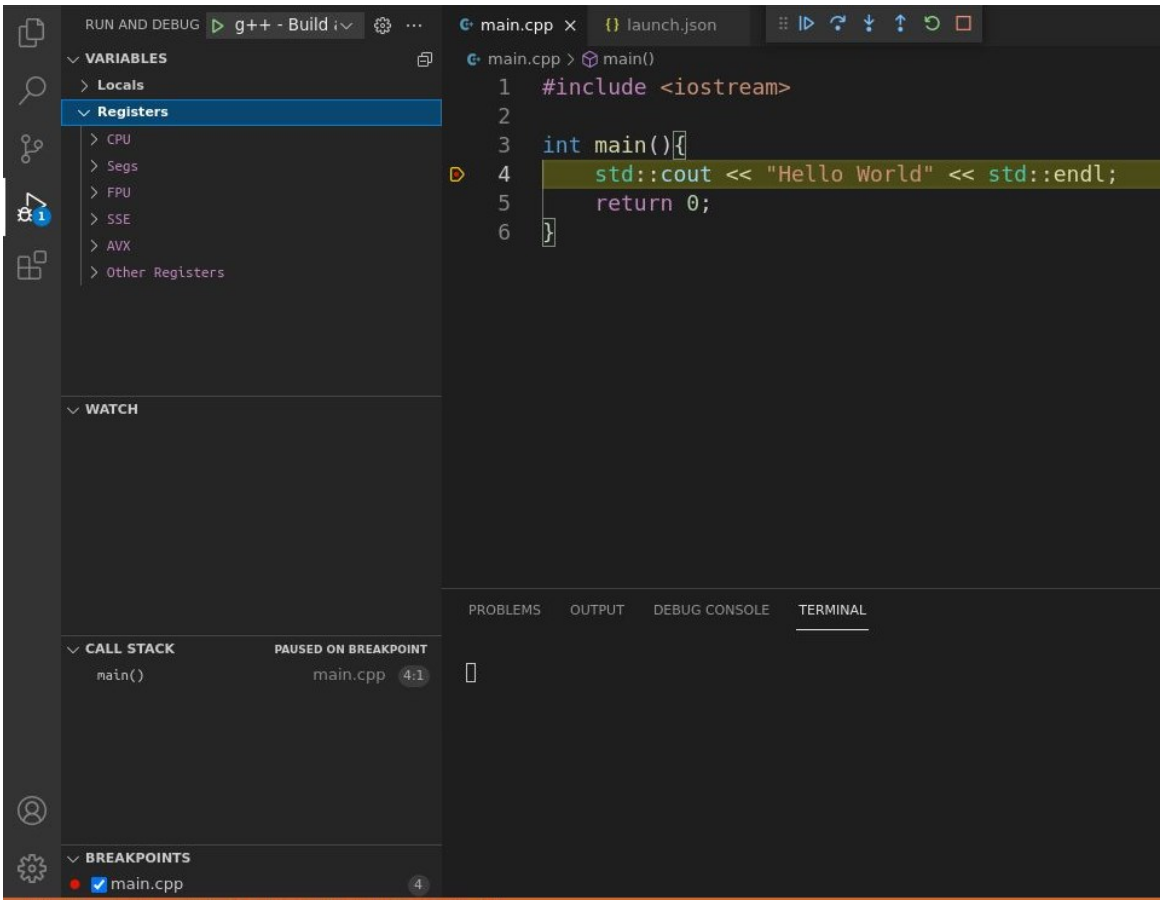
 Über Diesen Menüpunkt kann das Projekt gebaut, ausgeführt und debugged werden. Wurde dies für ein Projekt noch nicht durchgeführt, kann über die Schaltfläche ”Run and Debug” der Prozess initialisiert werden. Dafür kann ein Environment und anschließend eine Configuration ausgewählt werden. Diese sind für die Konfiguration des Run-/Debug-Prozesses von Nöten, sollten in der Regel jedoch von den Studenten nicht angepasst werden müssen. Über F5 kann jederzeit der Debugger gestartet werden. Sind keine dafür vorgesehenen Breakpoints (s. später) markiert, wird so das Programm ohne Unterbrechung ausgeführt. Alternativ kann über ”Run>Run Without Debugging” im oberen Menü auch direkt ohne Debugger ausgeführt werden.

Extensions

 Extensions erweitern VSCode um Tools und Funktionen. Dafür gibt es von VSCode einen Marketplace, über den Extensions heruntergeladen und installiert werden können bzw. installierte eingesehen und modifiziert oder gelöscht werden. In der dort gegebenen Searchbar kann nach Begriffen oder expliziten Namen einer Extension gesucht werden. Mit dem Tippen werden bereits unterschiedliche Vorschläge aufgelistet. Ein Klick auf eines der Elemente öffnet im Editor eine größere Übersicht über die ausgewählte Extension. Diese kann über einen kleinen blauen Button ”Install” in VSCode integriert werden.

4. Debugging

Wenn es zu Problemen im Code kommt, greifen viele meist zum Einfügen von Debuglogs. Dies ist zwar recht “schnell”, macht alles jedoch unübersichtlich und bietet nur begrenzt Nachverfolgbarkeit von Abläufen und Strukturen an. Durch Logs wird zudem viel hin und her geschoben im Code und interne Zustände müssen häufig zusätzlich in Variablen gespeichert werden. Das macht es besonders umständlich, wenn nicht klar ist, wo das Problem genau liegt. Abhilfe schafft dabei der Debugger (s. Bild auf nächster Seite). Er ermöglicht tiefe Einblicke in Register und Speicherzustände sowie die Nachverfolgbarkeit des Ausführungsprozesses durch Breakpoints. Der einfachste und häufigste Debugprozess soll hier kurz erläutert werden. Links neben den Zeilennummern im Editor können rote Punkte gesetzt bzw. entfernt werden. Dies sind sogenannte Breakpoints. Wird das Programm mittels Debugger ausgeführt, läuft der Code, bis er beim Breakpoint ankommt. Dort hält das Programm an und bietet Einsicht in aktuelle Zustände. Über Buttons in einer kleinen Leiste oben kann dann schrittweise der Code weiter durchlaufen werden, in den nächsten Funktionsteil gesprungen / übersprungen oder abgebrochen werden. Auf der linken Seite sind zudem Variablen in lokaler sowie in Registerform einsehbar, um alle internen Zustände gut zu überblicken. Eine Watch sowie der Call Stack sind auch dort zu finden.



Beispielhaftes Debugging

5. C/C++ Extension

Um VSCode C bzw. C++ Language Support zu ermöglichen, sollte die entsprechende Extension verwendet werden. Die Extension kann entweder über den erwähnten Extensions-Prozess installiert

oder mittels Konsole mit “code --install-extension ms-vscode.cpptools” integriert werden. Dies sollte in den zur Verfügung gestellten Systemen bereits erfolgt sein, gleiches gilt für das Aufsetzen der Runconfig. Die Extension bietet Crossplattform-Entwicklung für C sowie C++ und kommt mit einigen Tools. Obwohl Debuglogs eher unpraktisch sind, bietet die Extension eine spezielle Form von Logs, den sogenannten Logpoints. Sie bilden eine Art Mischung aus Debuglogs und Breakpoints. Mit einem Rechtsklick links der Zeilennummern (ähnlich wie Breakpoints) kann ein solcher Logpoint erstellt werden. Anders als Breakpoints hält der Code hier nicht an, bietet aber über Tokens eine Ausgabe zu unterschiedlichen Informationen wie z.B. Callstack, Funktion oder ähnlichem. Anders als normale Logs und Breakpoints sind diese darüber hinaus nicht Teil des Sourcecodes.

6. Shortcuts

In VSCode sind die meisten typischen Shortcuts beinhaltet. Über “File > Preferences > Keyboard Shortcuts” können diese eingesehen und angepasst werden. Neben den gängigsten bietet VSCode besonders auch Shortcuts zum Multcursor-Editing. Über Help kann sich zudem eine PDF generiert werden, die für das gegebene System die Shortcuts auflistet. Diese sind diesem Ordner neben dieser Anleitung beigelegt.