Seminararbeit: Eine Fallstudie zu eingebetteten Systemen mit Echtzeitanforderungen anhand von 3D-Druckern mit integriertem Webserver

Tobias Güthlein
Friedrich-Alexander-Universität Erlangen-Nürnberg

ZUSAMMENFASSUNG

In diesem Seminarpapier wird ein Überblick über die Echtzeitanforderungen im 3D-Druck gegeben und ein Papier von Cheng et al. betrachtet. Das Papier von Cheng et al. stellt *QduinoMC* vor, ein Multicore Ansatz für Arduinos und es wird verwendet, um einen internetverbundenen 3D-Drucker zu entwickeln. Ebenso wird in diesem Seminarpapier ein Überblick über *Qduino* und *QduinoMC* gegeben. Im Anschluss werden einige Punkte des internetverbundenen 3D Druckers diskutiert. Des Weiteren werden einige Aspekte bei den von Cheng et al. durchgeführten Experimenten mit *QduinoMC* diskutiert.

1 EINLEITUNG

Das Internet of Things wächst stetig an und umfasst viele Geräte wie z. B. Haushaltsgeräte von Smart Kühlschränken bis hin zu smarten Beleuchtungsanlagen, aber auch verschiedene Industriegeräte werden an das Internet angeschlossen, wie z. B. Fertigungsmaschinen oder Überwachungssysteme. Viele dieser Geräte verwenden Microcontrollern wie den Arduinos[1] für die Steuerung. Arduinos sind eine Open Source Mircocontroller-Plattform. Sie dienen dazu, elektronische Projekte zu steuern und bestehen aus einer Kombination von Hardware und Software. Dabei verfügen Arduinos über eine Reihe von verschiedenen Anschlüssen und Pins, die es ermöglichen, andere Komponenten anzuschließen und mit dem Arduino anzusteuern. So sind Arduinos in der Lage, beispielsweise Lichter, Motoren, Sensoren oder andere erdenkbare elektrische Geräte, die an die Arduinos angeschlossen werden können, zu kontrollieren. Des Weiteren können Arduinos programmiert werden, um komplexere Projekte zu realisieren. Dies geschieht, indem die Arduinos auf Eingaben reagieren und dementsprechend verschiedene Komponenten ansteuern. Ein Gerät, bei dem unter anderem Arduinos zur Steuerung verwendet werden, sind 3D-Drucker. Bei 3D-Druckern handelt es sich um Maschinen, die physische Objekte drucken können. Bei diesen Druckern wird beispielsweise Plastik oder Metall schichtweise aufgebaut, um ein Objekt zu erstellen [5]. Diese Fertigungsmethode wird z.B. für Hobbyprojekte wie das Erstellen von Figuren verwendet. Der 3D Druck ist außerdem ein nützliches Werkzeug, um schnell und billig Prototypen oder andere Objekte herzustellen. Der 3D-Drucker ist auch eine der Anwendungen, die zum Internet of Things beitragen kann. Deswegen wurde von Cheng et al. ein möglicher Ansatz erarbeitet, einen Webserver mit einem 3D-Drucker zu kombinieren. Bei der Steuerung von 3D-Druckern ist wichtig, dass es Echtzeitanforderung gibt, die eingehalten werden müssen, damit der 3D Druck funktioniert. Für diese Fälle haben Cheng et al. *QduinoMC* entwickelt, um Echtzeitanforderungen auf Multicore-Arduinos gerecht zu werden.

2 3D-DRUCKER UND IHRE ANFORDERUNGEN

2.1 Motorkontrolle

Damit ein 3D-Drucker überhaupt drucken kann, muss dessen Druckkopf in der richtigen Position sein. Es gibt verschiedene Möglichkeiten, den Druckkopf in die richtige Position zu bringen. Z. B. kann der Druckkopf selbst bewegt werden, die Druckplattform kann sich richtig zum Druckkopf ausrichten oder eine Kombination von beidem [6]. Jeder dieser Möglichkeiten benötigt Motoren, die die richtige Druckposition einstellen. Die meisten Drucker verwenden Schrittmotoren und Riemen. Das Drucken benötigt koordinierte Bewegungen [7], damit der Druckkopf zum richtigen Zeitpunkt an der richtigen Position ist. Damit der Motor sich flüssig bewegt, muss die zeitliche Verzögerung zwischen den Schrittmotorimpulsen genau berechnet werden. Diese Impulse geben dem Motor vor, einen Microschritt weiter zu laufen. Dies ist die Pulsfrequenz der Motoren. Falls sich die Generierung des Pulses durch zeitliche Störungen durch andere Berechnungen im Mikrocontroller verzögert, wird die Frequenz des Pulses abfallen. Dies kann zu einer variierenden Dicke der aktuellen Druckschrift führen, wodurch können Lücken im Druck entstehen [3]. In der Abbildung 1 sind diese Auswirkungen dargestellt. Dieser Effekt wird mit zunehmender Schichtanzahl größer und kann sich negativ auf die strukturelle Integrität des gedruckten Objektes auswirken [3]. Aus diesem Grund ist es bei 3D-Druckern sehr wichtig, dass Berechnungen nicht länger dauern als die veranschlagte Ausführungszeit.

2.2 Druckkopfkontrolle

Auch für den Druckkopf gibt es einige zeitkritische Berechnungen, wie z. B. die Temperatur Kontrolle des Druckmaterials. Damit der 3D-Druck auf Basis der Materialextrusion durchgeführt werden kann, muss das Druckmaterial erhitzt werden, um in einen semi-flüssigen Zustand zu gelangen [5]. Damit dieser Zustand gehalten werden kann, muss regelmäßig die Temperatur des Materials überprüft werden und gegebenenfalls angepasst werden. Der 3D-Drucker, der von Cheng et al. verwendet wurde, überprüft alle 8 ms die Temperatur durch einen Interrupt und passt diese über Ventilatoren oder eine Heizung an. Dies ist auch eine zeitkritische Aufgabe, da das Druckmaterial zu jedem Zeitpunkt während des Druckes im gewünschten, semiflüssigen Zustand sein muss.

3 QDUINO UND QDUINOMC

3.1 Qduino

Qduino ist ein Multi-Threaded Arduino System aufbauend auf dem Quest Open Source Echtzeit-Betriebssystem [2].

Abbildung 1: 2D Darstellung von variierenden Druckschichtdicken [3].

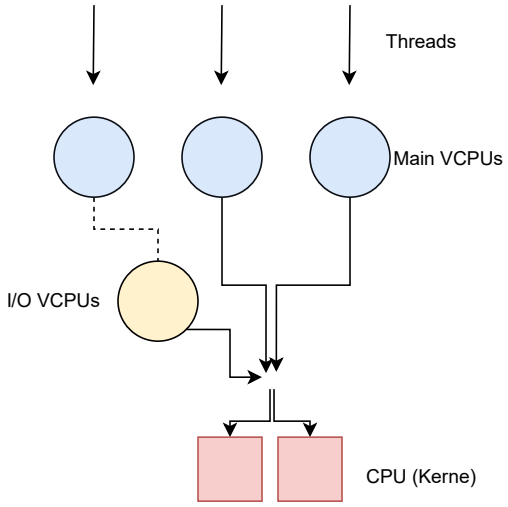

Abbildung 2: Leicht verkleinerte Abbildung des Quest VCPU-Schedulers [4].

Quest unterstützt Single und Multicore Prozessoren und bietet ein Echtzeit-Scheduling-Framework, bei dem Threads zu virtuellen CPUs (VCPUs) zugeteilt werden. Diese VCPUs dienen dazu, die Zeit zu erfassen, die von den Threads auf den CPUs verbracht wird und um die Ausführung von verschiedenen Threads zu priorisieren [4]. Der Scheduler stellt dabei eine zeitliche Isolation zwischen Systemevents und anderen Aufgaben sicher [2]. Hierfür wird bei Quest zwischen Main VCPUs und I/O VCPUs unterschieden. In Abbildung 2 wird diese Aufteilung dargestellt. Die Main VCPUs werden für normale Tasks verwendet und die I/O VCPUs werden für systemrelevante Aufgaben wie z. B. Interrupt-Handler von I/O Geräten verwendet. Das Quest Betriebssystem kann auf verschiedenen Plattformen benutzt werden, darunter auch Single-Board Computer wie z. B. MinnowMax oder Intel Galileo [4].

Mit *Qduino* lassen sich Arduinoprogramme im Quest Betriebssystem ausführen. Dabei kann ein Qduinoprogramm mehrere `loop()` Konstrukte verwenden. Hierzu wird jedem Loop ein separater Thread zugeteilt. Der Quest VCPU-Scheduler weist diese Threads den echten CPUs zu und gewährleistet durch ihn eine zeitliche Isolation der CPU-Ressourcen. Dadurch können mit *Qduino* parallelisierte Arduino-Anwendungen geschrieben werden [2]. Zusätzlich wurden noch Kommunikations- und Synchronisationsmechanismen für die Threads implementiert. Dazu zählen unter anderem die Kommunikation über globale Variablen und einem Ringbuffer, sowie die Synchronisation über Spinlocks [2].

Ebenso wurden die meistgenutzten Arduino API Funktionen sowie die Serial und Servo Bibliotheken in *Qduino* implementiert [2].

3.2 QduinoMC

QduinoMC ist eine Erweiterung von *Qduino* . Dabei wurde das Multi-Threaded Arduino System auf Multicore-Systeme erweitert. Bei *Qduino* konnte es zu Problemen kommen, wenn mehrere zeitkritische Loops sich denselben Kern mit dem Interrupt-Handlern teilen, während die anderen Kerne gerade mit nicht zeitkritischen Aufgaben beschäftigt sind. Somit wurde die Ausführung der zeitkritischen Loops behindert. Aus diesem Grund wurde von Cheng et al. *QduinoMC* entwickelt [3].

QduinoMC bietet eine Erweiterung der `loop()` Konstrukte, um diese an einen spezifischen Kern zu binden. Des Weiteren ist es mit *QduinoMC* ebenso möglich, den Kern dieser Loop nur für diesen einen Zweck zu verwenden, um Scheduling Overhead zu vermeiden. Ebenso ist es mit *QduinoMC* möglich, mehrere Loops mit jeweils eigenen Zeit-Budget im gleichen Programm zu erstellen. Dieses Programm aus mehreren Loops lässt sich ebenfalls durch *QduinoMC* in der Ausführung auf einen spezifizierten Kern begrenzen [3]. Durch diese Erweiterung lässt sich das Problem, dass nicht zeitkritische Aufgaben die Kerne belegen, beheben.

Das Festlegen von Aufgaben auf bestimmte Kerne ist in *QduinoMC* nicht nur auf Loops beschränkt. Ebenso können Interrupts auf bestimmte Kerne begrenzt werden, damit andere Zeit kritischen Aufgaben nicht durch diese gestört werden. Dies geschieht, indem die jeder Interrupt-Handler an seine eigne Quest I/O VCPU gebunden wird, optional kann hierzu auch noch ein bestimmter tatsächlicher Kern festgelegt werden. Durch ein Zeit-Budget der I/O VCPU wird somit verhindert, dass Interrupts die anderen zeitkritischen Aufgaben dauerhaft behindert werden [3]. In Abbildung 3 wird die Kernbindung beispielhaft an mehreren Loops dargestellt.

4 EXPERIMENTE

4.1 Beschreibung

Im Papier von Cheng et al. wurden zwei verschiedene Ansätze getestet, um einen Webserver mit einem 3D Drucker zu kombinieren. Dabei wurden jeweils 2 Experimente durchgeführt. Die Experimente bestanden beide aus zwei Fällen. Beim ersten Fall wurde der Webserver nicht eingeschaltet. Beim zweiten Fall wurde der Webserver einschaltet und unter Last gestellt. Dafür wurden durchgängig in zufälligen Zeitintervallen von 100 bis 1000 ms Aufträge

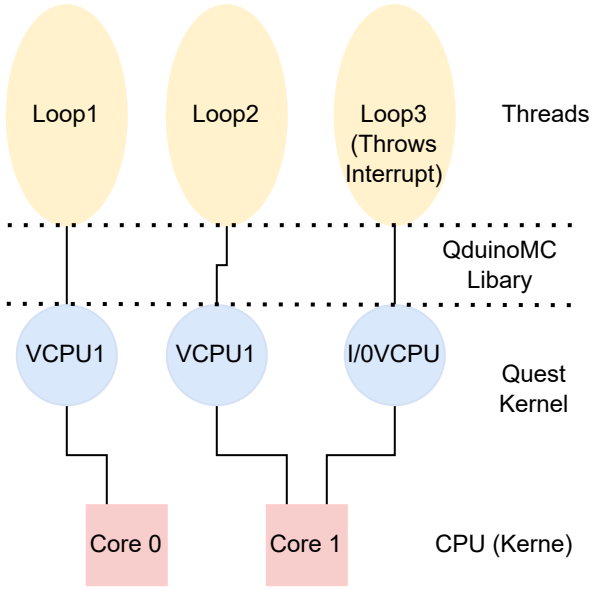

Abbildung 3: Beispielhafte Darstellung der Kernbindungen nach Marlin on *QduinoMC* [3].

an den Webserver geschickt. Die Aufträge hatten zufällig gewählte Dateigrößen im Bereich von 50 KB bis 150 MB. Experimente :

- **Experiment 1** Beim ersten Experiment ging es um die Echtzeit-Ausführung des Temperaturkontroll-Threads. Hierfür wurde die Temperatur im Druckkopf (Extruder) gemessen. Der Extruder wurde auf 209 Grad Celsius erhitzt und diese Temperatur sollte sich während des Druckvorgangs nicht verändern. Während des Druckens wurde die Temperatur immer wieder gemessen.
- **Experiment 2** Beim zweiten Experiment wurden die Auswirkungen des Webservers auf die Pulsfrequenz gemessen. Das Experiment untersuchte die Auswirkungen des Webservers auf die Motoren, wobei die Pulsfrequenz für den Antrieb der Motoren des Druckers relevant war. Während des Druckens wurde die Pulsfrequenz immer wieder gemessen. Dieses Experiment beabsichtigte die Generierung einer Pulsfrequenz von 10 kHz.

4.2 Beobachtungen bei der Linux-Implementation

Zum Vergleich zur QduinoMC Implementierung haben Cheng et al. eine Linux-Umgebung evaluiert. Für die Implementation eines mit dem Internet verbunden 3D-Druckers mittels Linux haben Cheng et al. den Mikrocontroller eines Printrbot Simple Metal 3D-Druckers ersetzt. Ersetzt wurde der Mikrocontroller mit einer Kombination aus einem Intel MinnowMax Board und einem RepRap Arduino Mega Polulo Shield. Auf dem MinnowMax Board wurde Yocto Linux 4.4.13 verwendet.

Bei der Implementation mit Linux wurde bei dem ersten Experiment von Cheng et al. festgestellt, dass die Temperatur gehalten wird, unabhängig vom aktuellen Fall. Nach einer kurzen Aufwärmphase wurde die Temperatur bei ausgeschaltetem Webserver bei circa 209 Grad Celsius gehalten. Es gab immer wieder kleine Abweichungen, aber diese lagen immer nahe an 209 Grad. Cheng et al. stellten Ähnliches fest, bei gestartetem Webserver. Auch bei diesem Experiment lagen die Temperaturen nach einer kurzen Aufwärmphase weitestgehend bei circa 209 Grad Celsius.

Bei dem zweiten Experiment mit der Linux-Implementation wurde von Cheng et al. festgestellt, dass die Pulsfrequenz bei ausgeschaltetem Webserver zwischen 7.75 und 8.02 kHz variierte. Es wurde also nicht die gewünschte Frequenz von 10 kHz erreicht. Bei eingeschaltetem Webserver wurde von Cheng et al. festgestellt, dass die Frequenz noch mehr variiert und einen noch niedrigeren minimal Wert von 6.42 kHz erreichte.

Cheng et al. identifizierten durch dieses Experiment zwei wesentliche Probleme mit dieser Implementierung. Das erste Problem besteht darin, dass das Frequenzintervall zu gering ist, um die Motoren auf ihre erwartete Geschwindigkeit zu bringen. Das zweite festgestellte Problem besteht darin, das der Kontrollthread für die Motoren von anderen Aufgaben gestört wird, was sich durch Zucken in der Motorrotation zeigte.

4.3 Beobachtungen bei der QduinoMC-Implementation

Die Experimente mit der Implementation eines mit dem Internet verbunden 3D-Druckers mittels *QduinoMC* wurden ebenfalls auf den Printrbot Simple Metal 3D-Drucker mit ersetztem Mikrocontroller aus der Linux-Implementation durchgeführt. Nur wurde hier *QduinoMC* anstatt von Linux verwendet.

Bei den Experimenten zu *QduinoMC* wurden von Cheng et al. auch Vergleichswerte erfasst, durch einen unveränderten Printrbot 3D-Drucker. Bei der Implementierung mit *QduinoMC* wurde beim ersten Experiment von Cheng et al. festgestellt, dass die Temperatur auch bei dieser Implementation in beiden Fällen bei 209 Grad Celsius gehalten wird. Ebenfalls wurde festgestellt, dass dies genauso effizient geschieht wie beim unveränderten Printrbot 3D-Drucker. Beim zweiten Experiment mit der QduinoMC-Implementierung wurde von Cheng et al. festgestellt, dass die Pulsfrequenz bei ausgeschaltetem Webserver stabil bei 9.569 kHz während des Experimentes blieb. Bei eingeschaltetem Webserver blieb die Frequenz auch stabil bei 9.569 kHz. Der unveränderte 3D-Drucker generierte eine stabile Pulsfrequenz von 9.96 kHz.

Aus diesen Experimenten wurde von Cheng et al. gefolgert, dass die Implementation mit *QduinoMC* zwar eine geringere Pulsfrequenz als der unveränderte 3D-Drucker aufweist, aber die Pulsfrequenz höher und stabiler ist als bei der Linux-Implementation.

5 DISKUSSION

In diesem Abschnitt werden das Thema und die durchgeführten Experimente aus dem Papier von Cheng et al. kritisch hinterfragt.

Es stellt sich die Frage, ob ein 3D-Drucker, der direkt mit dem Internet verbunden ist, überhaupt sinnvoll ist. Hier sollte man zwei verschiedene Fälle betrachten. Der erste Fall ist der Privatgebrauch von 3D-Drucker in Hobbyanwendungen und der zweite Fall ist der gewerbliche Gebrauch von 3D-Druckern in Fertigungsbetrieben (Printfarm). Des Weiteren wird die Durchführung der Experimente unter Serverlast kritisch hinterfragt. Ebenso werden zwei Aspekte zur Temperaturkontrolle und Laufzeit des Drucks dargelegt.

5.1 Hobbyanwendungen

Für Hobbyanwendungen ist das von Cheng et al. erwähnte Spooling von Druckaufträgen eher weniger nützlich. Bei vielen 3D-Druckern muss erst das letzte Werkstück aus dem Drucker entnommen werden, bevor der nächste Auftrag ausgeführt werden kann. Sofern kein Drucker mit automatischer Entnahme verwendet wird, was bei vielen Hobbyanwendern nicht der Fall sein wird, muss der Benutzer des 3D-Druckers vor dem Beginn des nächsten Druckauftrages selbst tätig werden und kann nicht den Drucker selbstständig weiter drucken lassen. Ein weiterer Grund, weshalb internetverbundene 3D-Drucker im Hobbygebrauch eher weniger nützlich sind, besteht darin, dass die Koordination verschiedener Drucker in den meisten Fällen nicht erforderlich ist. Da die meisten Hobbybenutzer nur einen Drucker haben werden, müssen sie nicht verschiedene Drucker untereinander koordinieren. Es gibt aber auch einen Aspekt für Internet verbundene 3D-Drucker im Hobbygebrauch. Dieser Aspekt ist, dass man den Druckstatus abfragen kann, obwohl man gerade nicht beim Drucker ist. Dieser Aspekt ist auch für Hobbyanwender von Vorteil, da ein Druck lange dauern kann und man so in der Lage ist den Drucker aus der Ferne zu überwachen. Zusammenfassend zeigt sich, dass die meisten Vorteile von Internet verbundenen 3D-Druckern im Hobbygebrauch ungenutzt bleiben.

5.2 Printfarm

Für den gewerblichen Gebrauch sind 3D-Drucker, die mit dem Internet verbunden sind, sehr nützlich. Wenn mehrere 3D-Drucker in einer Printfarm verwendet werden, können diese sich untereinander koordinieren. Zum Beispiel könnten Druckaufträge nur an Drucker geleitet werden, die über genügend Druckmaterial verfügen, um somit das manuelle Eingreifen, wie z. B. eine manuelle Zuweisung der Aufträge zu passenden Druckern oder das Auffüllen der Druckmaterialien zu minieren. Diese Koordination könnte auch noch mit dem Spooling verbunden werden, um die Druckvorgänge bei mehreren Druckern weiter zu optimieren. Indem Drucker, die gerade keinen Druckauftrag haben, offene Aufträge von anderen Druckern übernehmen, wenn diese gerade noch an anderen Aufträgen arbeiten. Auch beim gewerblichen Gebrauch von 3D-Druckern wäre eine Fernüberwachung der Drucker von Vorteil, da dadurch nicht nur das Personal der Printfarm den Druck überwachen können, sondern auch die Kunden über den Status ihrer Druckaufträge informiert bleiben. Aus diesen Gründen sind 3D-Drucker, die direkt mit dem Internet verbunden sind, sehr nützlich in Printfarmen.

5.3 Experimente mit unter Serverlast

In den Experimenten, die von Cheng et al. durchgeführt wurden, gab es den Fall, dass die Webserver, die auf den 3D-Drucker liefen, unter Last gestellt worden sind. Hierzu wurden bekamen die Server immer wieder in schneller Abfolge Dateien zwischen 50 KB und 150 MB zugesendet. Es wurde festgestellt, dass dies Auswirkungen auf eine der getesteten Implementationen hatte. Es ist zwar offensichtlich, dass es sich in diesem Fällen um einen Stresstest handelt, aber es stellt sich trotzdem die Frage, was soll dieser Stresstest genau darstellen? Im Normalfall würde kein Benutzer den eigenen gerade laufenden 3D-Drucker mit so vielen Anfragen behindern. Also bleibt nur die Möglichkeit, dass die 3D-Drucker von anderen Personen dauerhaft mit Aufträgen beliefert werden. Darin liegt eine der Schwachstellen des 3D-Druckers, der direkt mit dem Internet verbunden ist. Der Drucker kann von außerhalb behindert werden und ist somit nicht in der Lage, Aufträge zu erfüllen. Des Weiteren zeigten die Experimente, dass bei einer der Implementationen unter der Last die Motoren des Druckers nicht mit der benötigten Antriebsfrequenz versorgt wurde und dies zu zuckenden Motorbewegungen führte. Auf Dauer könnte dies zu Schäden am 3D-Drucker führen.

Diese beiden Gründe zeigen, dass es sinnvoll wäre, den Webserver nicht direkt auf dem Drucker zu haben und ihn eher auszulagern. Die Koordination von 3D-Druckern auf herkömmliche Weise über einen zentralen Steuercomputer hätte diese Schwachstelle nicht, da somit keine der vielen Anfragen an den Drucker gelangen, so lange der letzte Druckvorgang noch nicht beendet ist. Ebenso könnte man diesen Steuercomputer mit einem wirklichen Webserver kombinieren, um trotzdem noch die Vorteile Webservers, wie z. B. Fernüberwachung des Druckers zu nutzen.

5.4 Temperatur und Laufzeit

In allen Experimenten bei beiden Implementationen des 3D-Druckers mit integriertem Webserver hat sich die Temperatur des Druckkopfes nach einer anfänglichen Aufwärmphase nicht verändert. In allen Fällen zu jedem Zeitpunkt war die Temperatur ähnlich der des unveränderten 3D-Druckers. Hier wurde von Cheng et al. nicht weiter darauf eingegangen, warum dies so ist. Anscheinend war der Einfluss der Temperaturkontrolle zu gering, um einen Unterschied in beiden Implementierungen zu machen.

Ein weiterer interessanter Fakt, der nur nebenbei von Cheng et al. erwähnt wurde, besteht darin, dass die Implementierung mit *QduinoMC* fast doppelt so schnell beim Drucken war wie die Linux-Implementierung. Und dabei auch noch von besserer Qualität war. Dies lässt sich wahrscheinlich auf die instabile Pulsfrequenz der Linux-Implementation zurückführen. Die instabile Pulsfrequenz führte zu stockenden Motorbewegungen, was sich wiederum auf die Druckzeit und Qualität des Drucks ausgewirkt hat.

6 FAZIT

Insgesamt erweist sich *QduinoMC* als ein nützliches Werkzeug für Arduino-Projekte. Es unterstützt mehrere Kerne und kann sicherstellen, dass Echtzeitanforderungen von Arduino-Projekten eingehalten werden. Es ist vor allem bei komplexeren Projekten nützlich, da verschiedene Aufgaben parallelisiert werden und trotzdem Echtzeitanforderungen eingehalten werden können. Dies wurde von Cheng et al. gezeigt, indem ein 3D-Drucker mit einem integrierten Webserver entwickelt wurde. Cheng et al. hat durch Experimente gezeigt, dass die Implementierung des 3D-Druckers mit *QduinoMC* besser ist als eine Implementierung mit Linux. Des Weiteren wurden in diesem Seminarpapier einige Aspekte zur Nützlichkeit von 3D-Druckern, die direkt mit dem Internet verbunden sind, diskutiert. Die Vorteile dieser Drucker kommen am besten im gewerblichen Gebrauch in Printfarmen zur Geltung, da hier ihr volles Potenzial ausgenutzt werden kann. Es muss nur sichergestellt werden, dass die Drucker nicht von unbefugten Benutzern behindert werden können.

LITERATUR

[1] 2024. Arduino homepage. (Jan. 2024). https://www.arduino.cc/.

[2] Zhuoqun Cheng, Ye Li und Richard West. 2015. Qduino: a multithreaded arduino system for embedded computing. In *2015 IEEE Real-Time Systems Symposium*. IEEE, 261–272.

[3] Zhuoqun Cheng, Richard West und Ying Ye. 2017. Building real-time embedded applications on qduinomc: a web-connected 3d printer case study. In *2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 13–24.

[4] 2024. Quest operating system. (Jan. 2024). http://www.questos.org/.

[5] N. Shahrubudin, T.C. Lee und R. Ramlan. 2019. An overview on 3d printing technology: technological, materials, and applications. *Procedia Manufacturing*, 35, 1286–1296. DOI: https://doi.org/10.1016/j.promfg.2019.06.089.

[6] Mohamad Hasan Bin Tasneem und Gamal Talal Amer. 2019. Design, fabrication and testing of a 3d printer. In *Proceedings of the international conference on industrial engineering and operations management, Pilsen, Czech Republic*, 2334–2344.

[7] Benjamin Weiss, Duane W Storti und Mark A Ganter. 2015. Low-cost closed-loop control of a 3d printer gantry. *Rapid Prototyping Journal*, 21, 5, 482–490.