schlizbäda

B e d i e n u n g s a n l e i t u n g

EsploraGamingController

Gaming Controller basierend auf dem Arduino Esplora Board

Datum:
31.01.2019

Dieses Dokument wurde mit dem Textsatzsystem LaTeX erstellt

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Vielen Dank für Ihr Interesse an schlizbädas EsploraGamingController.
Hierbei handelt es sich um ein Steuergerät für PCs, das Tastatur- und Mausaktionen emulieren kann und den Steuergeräten (Gaming Controllern) der diversen Spielekonsolen nachempfunden ist. Es basiert auf dem Arduino-Board *Esplora*, das technisch einem *Arduino Leonardo* entspricht, aber bereits einige Sensoren und Aktoren enthält und dessen Leiterplatte grob die Form eines Gaming Controllers aufweist.

https://store.arduino.cc/arduino-esplora

1.1. Rechtliche Hinweise

Bei der Konzeption des EsploraGamingControllers wurde darauf geachtet, nur Software zu verwenden, die unter einer freien Lizenz wie GNU GPL oder Ähnlichem zur Verfügung gestellt wird oder ganz in die *Gemeinfreiheit* (engl. *public domain*) entlassen wurde.

Marken

Einige Bezeichnungen in dieser Schrift können Marken sein, deren Benutzung durch Dritte für deren Zwecke die Rechte der Inhaber verletzen können.

Links

In dieser Bedienungsanleitung sind Links zu externen Seiten im Internet enthalten. Diese Inhalte macht sich der Verfasser schlizbäda trotz Verlinkung nicht zu eigen, da sie nicht in seinem Einflussbereich stehen! Zum Zeitpunkt der Verlinkung waren keine rechtswidrigen Inhalte erkennnbar. Eine ständige Überprüfung auf etwaige rechtsverstoßende Änderungen ist dem Verfasser nach geltendem Recht nicht zuzumuten.
Sollten aktuelle oder künftige Inhalte jedoch rechtswidrig sein, so kann der Autor darüber per e-mail an mailto:himself@schlizbaeda.de informiert werden. Es werden dann entsprechende Maßnahmen zur Beseitigung des/der betroffenen Links ergriffen.

Lizensierung der Bestandteile des Projektes

Code:
Die meisten Code-Beispiele werden von Arduino in die Gemeinfreiheit (*public domain*) gestellt und können daher von jeder Person verwendet werden. Auch der Code für den Esplora-GamingController wurde aus den Codevorlagen
https://www.arduino.cc/en/Tutorial/EsploraJoystickMouse,
https://www.arduino.cc/en/Tutorial/EsploraKart und
https://github.com/circuit69/EsploraTinkerkit
entwicket. Aufgrund zahlreicher Weiterentwicklungen und Anpassungen beschloss der Autor jedoch, sein Werk unter der GNU GPL v3 zu veröffentlichen:
https://www.gnu.org/licenses/gpl-3.0.en.html

Gehäuse:
Bei der Internetsuche nach einem passenden Gehäuse wurde schließlich auf
https://www.thingiverse.com/thing:45880 ein Satz Dateien für einen 3D-Druck gefunden, der unter CC BY-NC 3.0 (http://creativecommons.org/licenses/by-nc/3.0/) lizensiert ist, also keine kommerzielle Nutzung gestattet.

Bildrechte

Alle inhaltlich relevanten Fotos und technischen Abbildungen in diesem Dokument stammen vom Verfasser schlizbäda selbst und werden hiermit von ihm unter der *Creative-Commons*-Lizenz **CC BY-SA 3.0** veröffentlicht. Sie dürfen daher von jedem bei Namensnennung des Urhebers in unveränderter oder auch in veränderter Form unter den gleichen Bedingungen weitergegeben werden:

Das in dieser Dokumentation verwendete Arduino-Logo ist gemeinfrei:

1.2. Hinweis zur neumodernen „Genderei“ landauf, landab

Der Autor schlizbäda kann gar nicht oft genug betonen, dass ihm eine Gleichbehandlung **aller** Geschlechter (mittlerweile mehr als zwei) äußerst wichtig ist. Er verurteilt eine Diskriminierung von Menschen nur aufgrund ihres Geschlechtes oder anderer Nebensächlichkeiten wie ihrer Herkunft etc. aufs Schärfste!

Dennoch – oder gerade deswegen – lehnt er die derzeit grassierende Unart des sogenannten „Genderns“ mit sprachlichen Auswüchsen wie „AnwenderInnen“ oder gar „Benutzer*innen“ (jetzt neu mit Stern!) zugunsten einer klaren und verständlichen Ausdrucksweise ab. In diesem Trend sind selbst Wörter wie „Abiturienten“ verpönt: Soll man hier wirklich gender-gerecht „Abiturierende“ schreiben? 

1.3. Danksagung

schlizbäda möchte dem Benutzer **@dale** (https://forum-raspberrypi.de/user/23726-dale/) aus dem deutschen Raspberry Pi Forum (https://forum-raspberrypi.de) seinen Dank aussprechen: Er erklärte sich freundlicherweise bereit, das Esplora-Gehäuse von https://www.thingiverse.com mit seinem 3D-Drucker auszudrucken.

1.4. Kurzbeschreibung

Der EsploraGamingController ist ein den Gaming Controllern der gängigen Spielekonsolen nachempfundenes Steuergerät mit USB-Anschluss. Anstelle von proprietären Signalen übermittelt dieses Gerät bei Betätigung der Bedienelemente entsprechende Tastatur- bzw. Mauskommandos, um damit die laufende Software auf dem Computer zu steuern, an dem es über USB2.0 angeschlossen ist.

Aufgrund seiner quelloffenen und freien Programmierung kann es auf beliebige Anwendungen/Szenarien angepasst werden, indem *im Quellcode* den einzelnen Bedienelementen (Sensoren) die gewünschten Tastaturcodes oder Mauskommandos zugeordnet werden.

Es können acht verschiedene Belegungen für unterschiedliche Anwendungsfälle hinterlegt werden, die im laufenden Betrieb umgeschaltet werden können. Die gewählte Einstellung wird über die Farbe der auf dem Esplora-Board verbauten dreifarbigen LED angezeigt.

Softwareseitig emuliert der EsploraGamingController eine USB-Tastatur und eine USB-Maus und kann auf den meisten PC-Betriebssystemen (Windows, GNU/Linux) sowie auf einem Raspberry Pi unter Raspbian sofort ohne Treiberinstallation verwendet werden.

2. Bedienungsanleitung

Dieses Kapitel ist eine klassische Bedienungsanleitung für Benutzer, die den EsploraGaming-Controller mit den aktuell implementierten Funktionen verwenden wollen, ohne dabei die Software oder Hardware des Gerätes zu verändern.

2.1. Bedienelemente des EsploraGamingControllers

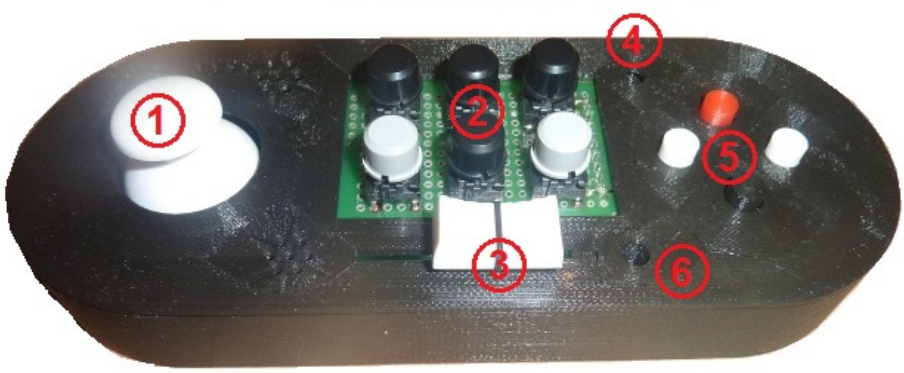


Abbildung 2.1.: *Bedienelemente (Sensoren) des EsploraGamingControllers*

Die Bedienung von Tastatur und Maus über den EsploraGamingController erfolgt über folgende Bedienelemente:

Nr.	**Element**	**übliche Funktionalität**
1)	Joystick	Mausbewegung
2)	TinkerKit-Tasten	Maustasten, Tastatur
3)	Schieberegler	Maus-Scrollrad
4)	Helligkeitssensor	*meist deaktiviert*
5)	Cursortasten	Tastencodes für Bewegung
6)	dreifarbige LED	Anzeige der ausgewählten Belegung

Tabelle 2.1.: *Bedienelemente (Sensoren) des EsploraGamingControllers*

Joystick

Der Joystick emuliert in den meisten Fällen Mausbewegungen. Je nach Auslenkung des Joysticks wird die Geschwindigkeit der Mausbewegung angepasst. Durch Druck auf den Joystick wird ein Klick auf die linke Maustaste emuliert.
Auf dem Esplora-Board ist der Joystick über zwei Potentiometer umgesetzt, deren Stellung über A/D-Wandler des verbauten 8-bit ATMEL-Mikrocontrollers *ATmega32U4* eingelesen werden. Bei Druck auf den Joystick wird ein Drucktaster betätigt, der digital eingelesen wird.

TinkerKit-Tasten

Auf dem EsploraGamingController ist eine kleine Zusatzplatine an der Stelle verbaut, an der eigentlich ein kleines Display vorgesehen wäre. Auf dieser Zusatzplatine befinden sich sechs Drucktaster, die mit den vier TinkerKit-Anschlüssen verbunden sind. Die weißen TinkerKit-Anschlüsse sind als Analogeingänge realisiert, so dass über eine geschickte Spannungsteilerschaltung unterschieden werden kann, welche Tastenkombination betätigt ist.
Diese sechs Taster sind mit zusätzlichen Tastaturcodes oder Mausklicks belegt.

Schieberegler

Der Schieberegler ist ein Potentiometer, das an einem A/D-Wandler des ATMEL-Mikrocontrollers angeschlossen ist. Hiermit wird das Scrollrad der Maus emuliert.
Bei der Bedienung reicht das Loslassen des Reglers natürlich nicht, um das emulierte Scrollrad zu stoppen! Vielmehr muss zum Anhalten des Scrollvorgangs der Regler kurz in die Gegenrichtung geschoben werden.

Helligkeitssensor

Auch der Helligkeitssensor ist an einem A/D-Wandler des ATMEL-Mikrocontrollers angeschlossen. Trotz einiger Versuche konnte der Helligkeitssensor nicht so implementiert werden, dass er wirklich eindeutige Signale im Sinne einer Betätigung liefert. Durch leichte Abschattungen entstehen Fehlinterpretationen. Dennoch ist er nach wie vor im Arduino-Sketch als Bedienelement enthalten. Er ist allerdings in den meisten Einstellungen deaktiviert.

Cursortasten

Hierbei handelt es sich um vier Drucktaster, die vom ATMEL-Mikrocontroller digital eingelesen werden. Sie emulieren in der Regel Tastaturbefehle für die Cursorsteuerung oder zur Bewegung einer Spielfigur, was in vielen Computerspielen mit den Tasten A, W, S und D erfolgt.

dreifarbige LED

Die dreifarbige LED dient als einfache Anzeige der ausgewählten Belegung, siehe Abschnitt 2.2

2.2. Belegung der Bedienelemente auswählen

Durch gleichzeitiges Betätigen **aller vier Cursortasten** wird in den Einstellmodus des EsploraGamingControllers gewechselt. Anschließend können mit den Cursortasten OBEN und UNTEN die Belegungen der Steuerelemente geändert werden. Mit jedem Druck ändert sich die Farbe der dreifarbigen LED entsprechend der aktuell gewählten Einstellung.
Durch erneutes Betätigen aller vier Cursortasten wird der Einstellmodus wieder verlassen. Um beim Verlassen ein versehentliches Ändern der Einstellung zu verhindern, wird empfohlen, beim Betätigen mit den Cursortasten LINKS oder RECHTS zu beginnen.

Nr.	**LED-Farbe**	**Bezeichnung**
0)	*„dunkelweiß“*	common 1
1)	rot	MinecraftPi
2)	grün	yamuplay *t.b.d. (to be defined)*
3)	gelb	«undefined 3»
4)	blau	«undefined 4»
5)	violett	«undefined 5»
6)	türkis	«undefined 6»
7)	weiß	common 2

Tabelle 2.2.: *Die acht Belegungen der Bedienelemente des EsploraGamingControllers*

common 1: LED: „dunkelweiß“ (sie leuchtet nur sehr schwach)
In dieser Einstellung emulieren die Bedienelemente ausschließlich Tastatureingaben. Es sind keine Mausfunktionen aktiv:

Joystick	Cursortasten
Joystick Druck	ENTER-Taste
Schieberegler	Bild-Auf, Bild-Ab
Helligkeitssensor	*deaktiviert*
TinkerKit orange	Leertaste, ESC-Taste
TinkerKit weiß	SHIFT, CTRL, ALT

MinecraftPi: LED: rot

Die Belegung ist auf das Spiel *MinecraftPi* für den Raspberry Pi ausgelegt.

Joystick	Mausbewegung zur Einstellung der Blickrichtung
Joystick Druck	Taste `E` zum Öffnen des Block-Menüs
Schieberegler	Scrollrad der Maus zur Schnellauswahl von Blöcken
Helligkeitssensor	*deaktiviert*
TinkerKit orange	linke und rechte Maustasten
TinkerKit weiß	ENTER, SHIFT, ESC, Leertaste

yamuplay: LED: grün

Hier ist geplant, eine Belegung zur Steuerung von schlizbädas Mediaplayer *yamuplay.py* zu implementieren.
https://github.com/schlizbaeda/yamuplay

Die hinterlegte Belegung ist derzeit undefiniert!

«undefined 3»: LED: gelb

Die hinterlegte Belegung ist derzeit undefiniert!

«undefined 4»: LED: blau

Die hinterlegte Belegung ist derzeit undefiniert!

«undefined 5»: LED: violett

Die hinterlegte Belegung ist derzeit undefiniert!

«undefined 6»: LED: türkis

Die hinterlegte Belegung ist derzeit undefiniert!

common 2: LED: weiß

In dieser Einstellung emulieren die Bedienelemente Tastatur- und Mauseingaben:

Joystick	Mausbewegung
Joystick Druck	Maus: Linksklick
Schieberegler	Maus: Scrollrad
Helligkeitssensor	*deaktiviert*
TinkerKit orange	linke und rechte Maustasten
TinkerKit weiß	SHIFT, CTRL, ALT

3. Änderung am EsploraGamingController

In diesem Kapitel wird die Herangehensweise für Änderungen an diesem Projekt beschrieben: Der Hardwareteil beschreibt in erster Linie den Einbau der ergänzten Zusatzleiterplatte mit den Tastern für die TinkerKit-Anschlüsse.
Für den Softwareteil wird vorausgesetzt, dass der Leser die Arduino-IDE erfolgreich auf seinem PC installiert hat und weiß, wie eine Arduino-Baugruppe damit programmiert wird.

3.1. Beschreibung der Zusatzplatine für die TinkerKit-Taster

Moderne Gaming Controller haben in der Regel mehr als nur die vier Cursortasten. Meist sind sogenannte Schultertasten an der Vorderkante des Controllers verbaut, die direkt mit den Zeigefingern bedient werden können.

Beim EsploraGamingController werden die vier TinkerKit-Anschlüsse verwendet, um weitere Taster hinzuzufügen. Da die weißen TinkerKit-Anschlüsse auf Pins des *ATmega32U4*-Mikrocontrollers verdrahtet sind, die über einen 10bit A/D-Wandler eingelesen werden, können pro weißem TinkerKit-Anschluss mehrere Taster über entsprechende Spannungsteiler **gleichzeitig** betrieben und unterschieden werden.

Auf dem ersten Prototypen der Zusatzplatine wurden an jedem analogen TinkerKit-Anschluss (weiß) jeweils zwei Taster mit den Spannungsteilern 33k : 22k bzw. 33k : 47k angeschlossen. Dies resultiert in folgenden Spannungen und A/D-Werten:

Taster	**A/D-Wert**	**Spannung**
kein Taster	1023	5,00V
47k-Taster	601	2,94V
22k-Taster	410	2,00V
beide Taster	320	1,56V

Tabelle 3.1.: *A/D-Werte der TinkerKit-Taster*

Die folgenden Bilder zeigen den Anschluss des Prototyps. Im Anhang ist das Schaltungskonzept sowie eine Leiterplatte dargestellt, die bis zu sechs Taster an den weißen TinkerKit-Anschlüssen ermöglichen.

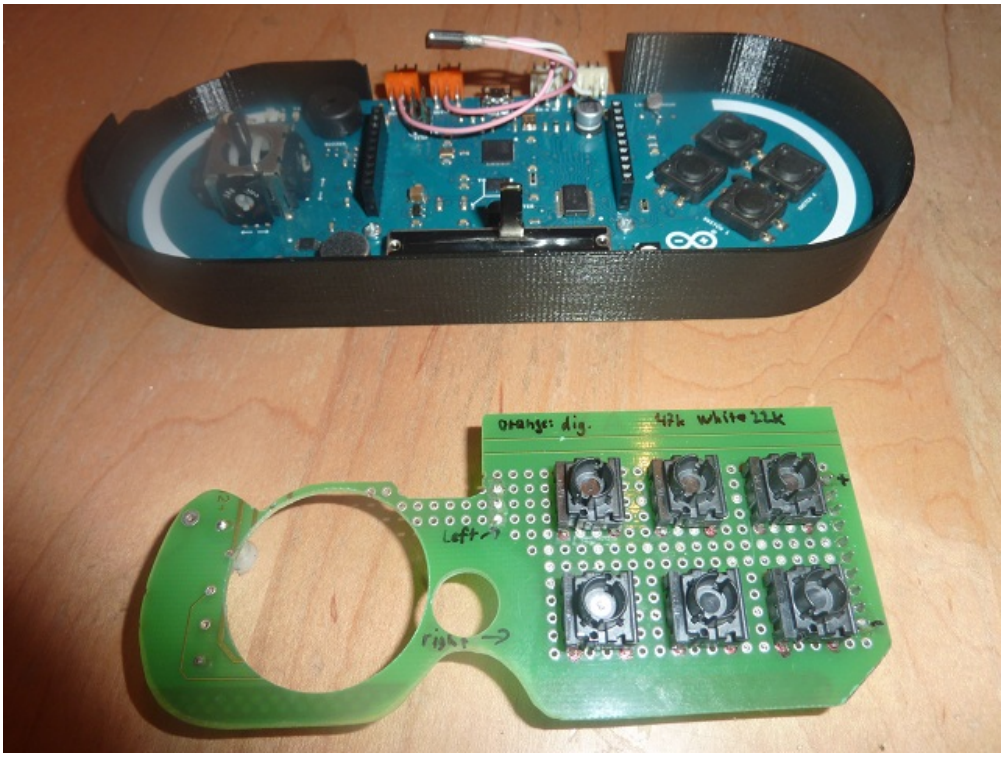


Abbildung 3.1.: *Arduino Esplora und TinkerKit-Zusatzplatine (Prototyp)*

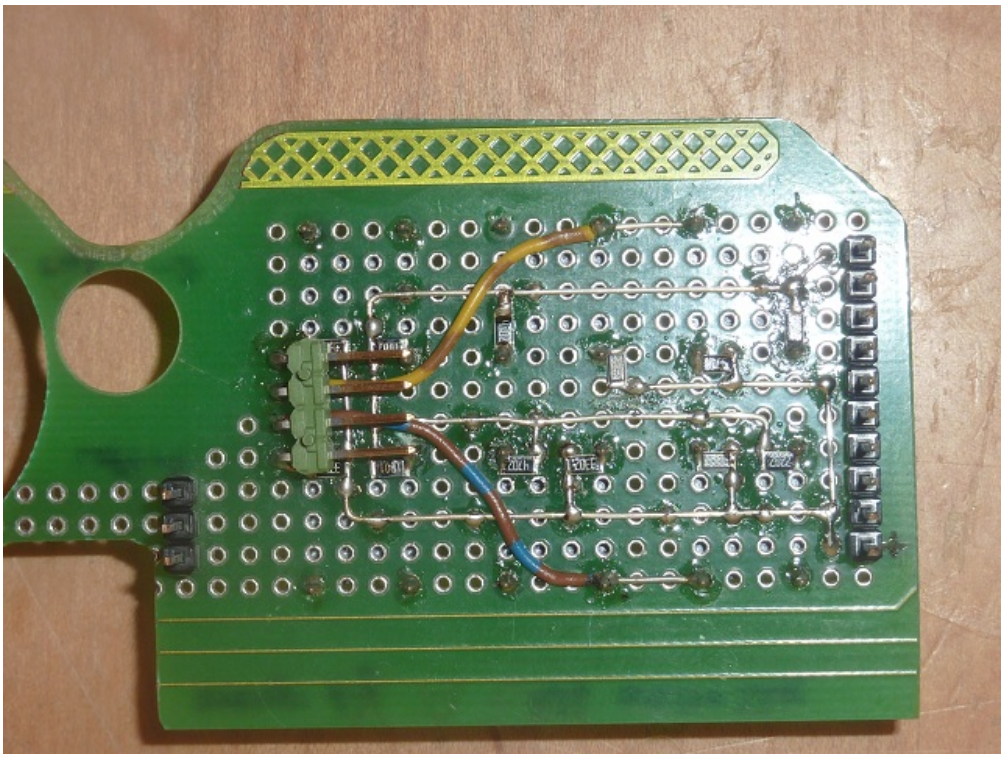


Abbildung 3.2.: *Widerstände der TinkerKit-Zusatzplatine*

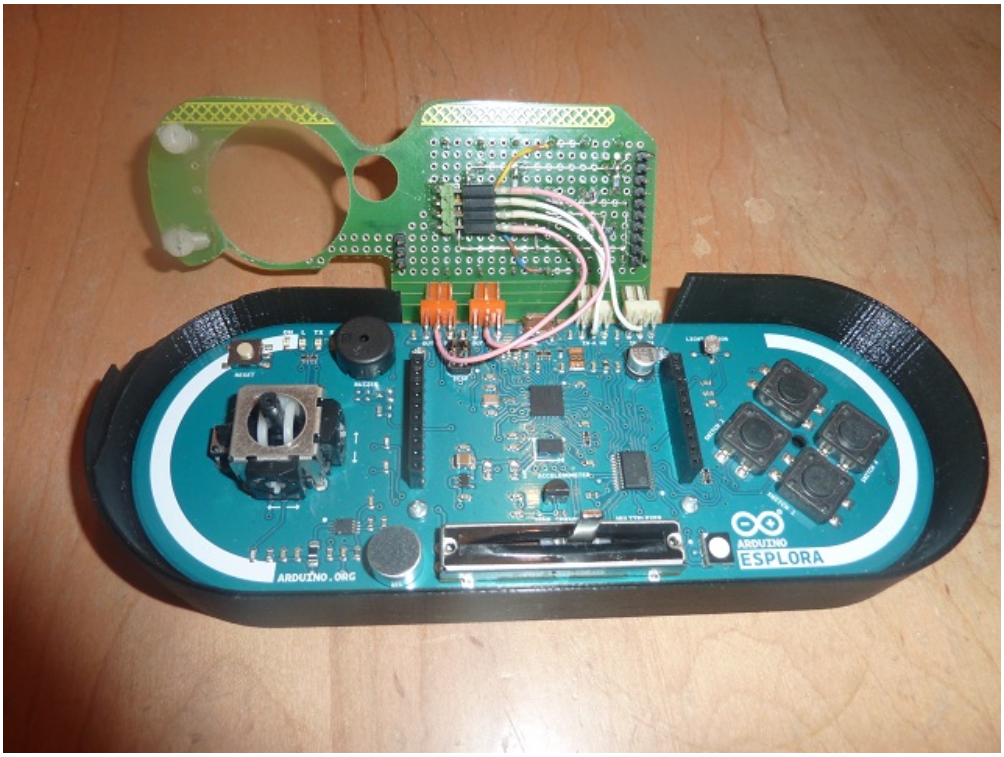


Abbildung 3.3.: *Anschluss der TinkerKit-Zusatzplatine am Arduino Esplora*

Abbildung 3.4.: *TinkerKit-Zusatzplatine eingebaut*

3.2. Beschreibung des Arduino-Sketch für den EsploraGamingController

3.2.1. Installation der Arduino-IDE

Eine vollständige Beschreibung zur Inbetriebnahme des *Arduino Web Editors* oder der *Arduino Desktop IDE* findet sich auf der Internetseite https://www.arduino.cc/en/Guide/HomePage. Dabei sollte sich jeder Anwender nach eigenem Gusto für eine Variante entscheiden.

3.2.2. Anpassung des Codes

Ein Arduino Esplora Board, das mit dem heruntegeladenen Sketch `EsploraGamingController.ino` programmiert wird, emuliert eine USB-Maus und eine USB-Tastatur, wie in Kapitel 2 beschrieben.

Die einfachste Art, den Code anzupassen, ist, die Werte der zweidimensionalen Arrayvariablen `gl_modeKeycodes` den eigenen Wünschen gemäß abzuändern und das Arduino Esplora Board neu zu flashen. Eine Liste der Tastaturcodes für nicht-ASCII-Zeichen befindet sich in der Datei `Esplora_ExtendedKeycodes.txt`.
Der Wert 0 bedeutet dabei, dass die im Code vorgesehene **Mausbelegung** für das entsprechende Bedienelement verwendet wird:

Bedienelement	**Mausbelegung bei Wert 0**
Joystickbewegung	Mausbewegung
Joystick Druck	Linksklick
TinkerKit orange	Linksklick, Rechtsklick
TinkerKit weiß	**keine Mausfunktionalität!**
Schieberegler	Scrollrad
Helligkeitssensor	Rechtsklick
Cursortasten	**keine Mausfunktionalität!**

Tabelle 3.2.: *Standardmausbelegungen der Bedienelemente*

Um ein Bedienelement ganz zu deaktivieren, muss ihm der Wert 255 (bzw. die Konstante `FUNCT_DISABLED`) zugewiesen werden.

Die analogen Bedienelemente liefern einen Wert zwischen 0 und 1023, da sie über einen 10-bit A/D-Wandler eingelesen werden. Beim Joystick bestimmen die ausgelesenen 10-bit-Werte die Geschwindigkeit der Mausbewegung. Gleiches gilt bei der Definition von Tastaturcodes für den Joystick: Hier wird die Wiederholrate der entsprechenden Taste ebenfalls von der Joystickauslenkung bestimmt.
Der Schieberegler emuliert die Drehung des Scrollrades der Maus. Auch hier bestimmt die Auslenkung die Geschwindigkeit der Scrollbefehle. Um das Scrolling zu stoppen, muss der Schieberegler nur leicht zurück in Richtung Mitte geschoben werden. Dies funktioniert sogar dann, wenn der ausgelesene Wert an sich ein Scrolling bewirken würde. Relevant ist hier das leichte Zurückschieben des Reglers.
Bei den weißen TinkerKit-Tasten wird der Analogeingang dafür verwendet, eine Kombination aus jeweils zwei Tasten pro TinkerKit-Anschluss zu ermitteln. Dies wird auf der Zusatzleiterplatte durch eine passende Kombination von Spannungsteilern ermöglicht. Die Genzwerte sind in der Arrayvariablen `TINKERKIT_LEVEL` definiert. Dabei ist darauf zu achten, dass die angegebenen Grenzwerte die Mittelwerte der jeweiligen Tastendrucke sind, um eine sichere Auswertung zu gewährleisten.
Der Helligkeitssensor liefert ebenfalls einen 10-bit-Wert. Allerdings kann er nur schwer für Betätigungsaktionen verwendet werden: Bereits kleine Änderungen der Beleuchtungsverhältnisse, die durch versehentliche Abdeckung des Sensors oder Drehen des ganzen Controllers verursacht werden, führen zu fehlerhaften Aktionen. Daher wird empfohlen, den Helligkeitssensor nicht zu verwenden und zu deaktivieren.

3.2.3. Aufbau des Codes

In diesem Abschnitt wird auf die wichtigsten `#define`s, Variablen und Funktionen des Quellcodes eingegangen:

serielle Schnittstelle (UART)

Dieser Sketch emuliert neben Tastatur und Maus auch eine USB-Schnittstelle mit der Baudrate 9600. Unter Windows ist diese serielle Schnittstelle im Geräte-Manager als ***COM<x>*** sichtbar, unter GNU/Linux als `/dev/ttyACM<x>`. Derzeit sind die Daten, die über diese serielle Schnittstelle zum PC gesendet werden, noch nicht sauber strukturiert!

UART vernünftig umsetzen

`#define`

`#define SERIAL_DEBUG`
Aktivieren/Deaktivieren von Debugmeldungen über die serielle Schnittstelle.

`#define LOOP_DELAY`
Legt die Wartezeit bei jedem Durchlauf in Millisekunden fest. Je kleiner dieser Wert ist, umso schneller reagiert das Arduino Esplora Board auf die Änderungen. Dies macht sich vor allem bei der Geschwindigkeit des Mauszeigers aufgrund von Joystickbewegungen bemerkbar.
Zu kurze Zeiten wirken sich aufgrund prellender Tasten möglicherweise negativ auf das Verhalten des Boards aus: Gedrückte Tasten werden dann mitunter doppelt ausgeführt!

Variablen

Variable `byte gl_mode`
Enthält die aktuell ausgewählte Belegung als Zahlenwert zwischen 0 und 7.

Arrayvariable `char* gl_modeNames[8]`
Hier sind Bezeichnungen für die acht unterschiedlichen Belegungen hinterlegt. Diese Strings werden bei der Auswahl über die serielle Schnittstelle ausgegeben.

zweidimensionale Arrayvariable `char gl_modeKeycodes[8][18]`
Hier werden die Tastatur- / Mauskommandos für die einzelnen Bedienelemente eingetragen:

0	Mauskommando
1-254	Tastaturkommando
255	deaktiviert

Funktionen

Funktion `void setup()`
Dies ist eine Standardfunktion eines Arduino-Sketches, die über die Arduino-Bibliothek eingebunden wird. Sie wird zu Beginn einmalig ausgeführt: Initialisierung der Variablen und Kalibrierung der analogen Sensoren, insbesondere des Joysticks.

Funktion `void loop()`
Dies ist eine Standardfunktion eines Arduino-Sketches, die über die Arduino-Bibliothek eingebunden wird. Sie wird nach `setup()` wiederholt in einer Endlosschleife aufgerufen und dient dem Pollen der Sensoren der Bedienelemente.

Funktion `void setMode(byte mode)`
Wählt die angegebene Belegung (0-7) aus:

- Setzt die Variable `gl_mode` auf den parametrierten Wert
- Setzt die Farbe der dreifarbigen LED
- gibt die Bezeichnung der parametrierten Belegung über die serielle Schnittstelle aus.

Funktion `unsigned int readTinkerkitInput(byte whichInput)`
Einlesen der TinkerKit-Eingänge über Aufruf der Funktion `readChannel(byte channel)`.

Funktion `unsigned int readChannel(byte channel)`
Diese Funktion überschreibt die gleichlautende Funktion in der Arduino-Library `Esplora`. Das Auslesen der TinkerKit-Signale erfolgt über einen Multiplexer-Chip auf dem Arduino Esplora Board.

Funktion `byte tinkerkitWhiteSwitches(unsigned int analogue)`
Auswertung der gedrückten Tastenkombination an den weißen TinkerKit-Anschlüssen. Hier ist die Variable `const unsigned int TINKERKIT_LEVEL[4]` definiert, in der die analogen Grenzwerte für die Tastenkombinationen hinterlegt werden. Man findet diese Grenzwerte am besten heraus, indem man diese Funktion mit allen Tastenkombinationen aufruft und die Messwerte über die serielle Schnittstelle ausgibt. Diese Messwerte sollten ziemlich genau in der Mitte zwischen zwei in dieser Arrayvariablen hinterlegten Grenzwerten liegen. Oder andersherum gedacht, sollten die Grenzwerte in der Mitte der Messwerte von zwei „benachbarten“ Tastenkombinationen liegen.
Damit diese Funktion vernünftig funktionieren kann, müssen die Widerstände für die Spannungsteiler auf der Zusatzleiterplatte richtig dimensioniert worden sein!

3.3. Ausblick auf geplante Erweiterungen

1. Änderung der Tastenbelegung durch den Anwender über das Gerät selbst, ohne Programmierung über die Arduino IDE
2. Speichern der Tastenbelegung im EEPROM des *ATmega32U4*
3. Definition mehrerer Tastaturbefehle als Sequenz pro Sensor
4. Definition von Werten für bestimmte Mausaktionen
5. vernünftige Ausgaben für die serielle Schnittstelle festlegen

A. Anhang

Der Anhang enthält die EAGLE-Zeichnungen zur TinkerKit-Zusatzplatine:

- Verdrahtungskonzept der TinkerKit-Taster
- Schaltplan der TinkerKit-Zusatzplatine
- Leiterplatte
  - Abmessungen
  - Oberseite
  - Unterseite

A.1. Bestückvarianten

Die Leiterplatte kann so bestückt werden, dass sie funktional dem in Abschnitt 3.1 beschriebenen Prototypen entspricht.
Alternativ kann die Bestückung für sechs über Spannungsteiler unterscheidbare Taster an den analogen Eingängen der weißen TinkerKit-Anschlüsse ausgeführt werden. Die Taster für die digitalen TinkerKit-Anschlüsse (orange) können an anderer Stelle (z. B. als Schultertasten am Gehäuse) verbaut werden und an der Stiftleiste X1 angeschlossen werden.

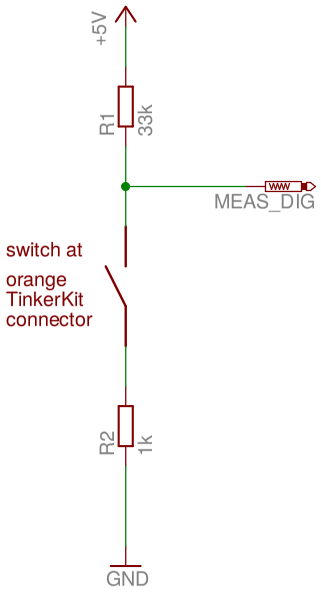
Die gewünschte Version kann über die dazugehörige Bestückvariante aus Tabelle A.1 festgelegt werden:

Version	**RDIG_ORL, RDIG_ORR**	**RANA_ORL, RANA_ORR**	**RDN_ORL, RDN_ORR**	**RDN_PINL, RDN_PINR**
vier analoge Taster, zwei digitale Taster	0R	nicht bestückt	1k	nicht bestückt
sechs analoge Taster, zwei externe Taster	nicht bestückt	0R	68k oder 82k (t.b.d.)	1k

Tabelle A.1.: *Bestückvarianten*

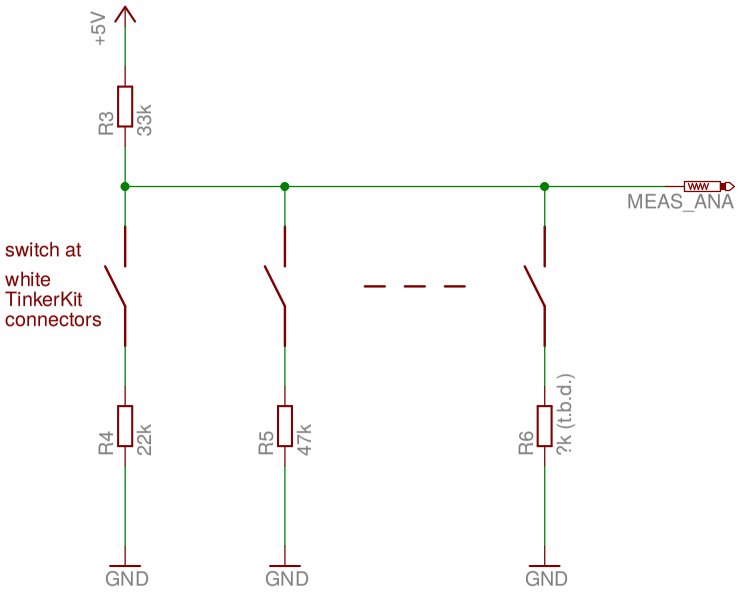
Digital Input

at orange TinkerKit connectors



Analogue Inputs

at white TinkerKit connectors



Measurement

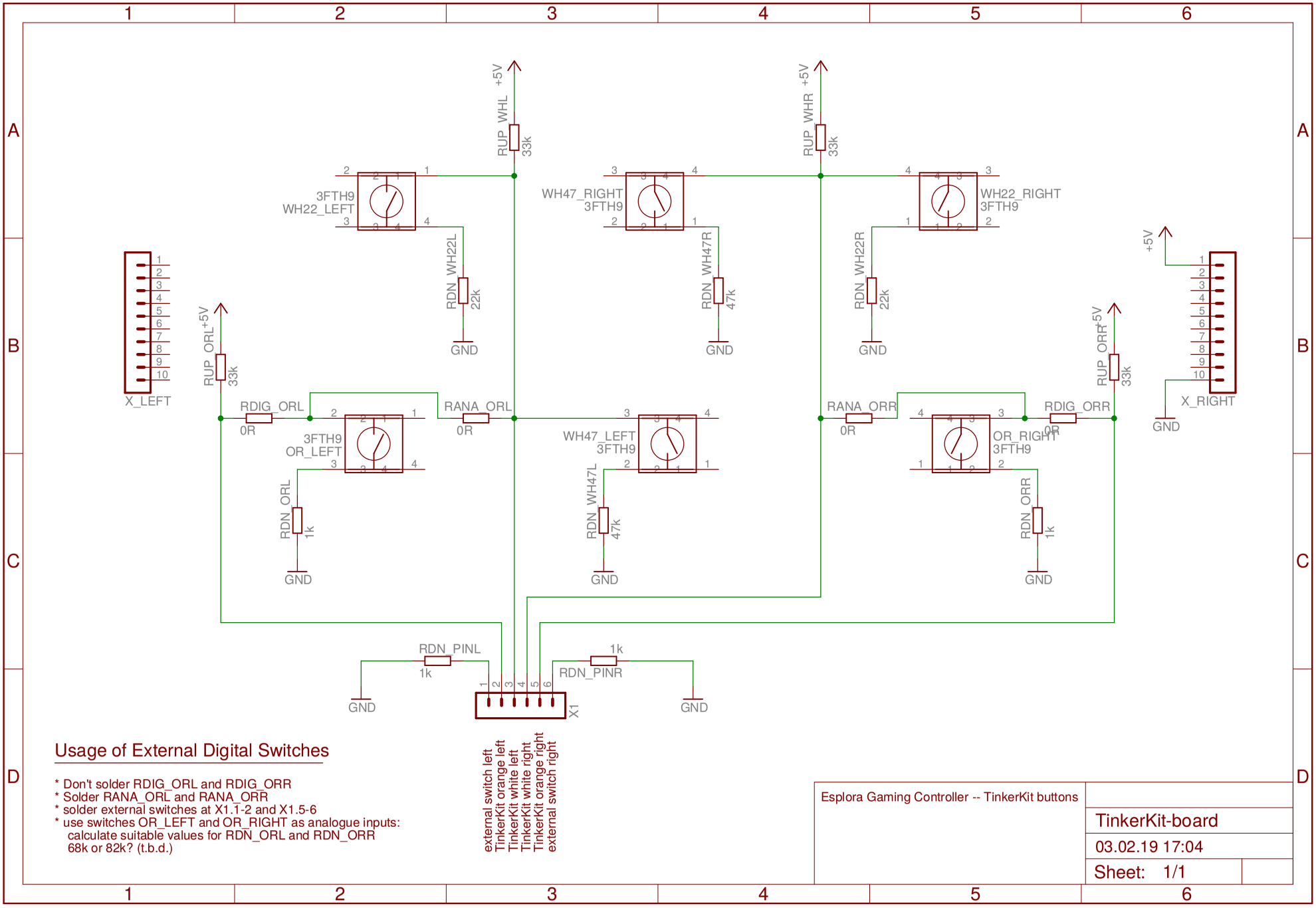
MEAS_ANA is connected to an analogue input
of the ATMEL ATmega32U4 micro controller.
There is is a 10 bit A/D converter in the background
which results in a resolution fom 0 to 1023.

A sophisticated combination of different voltage dividers
gives the possibility to recognize which switches are pressed.

Even a combination of several switches can be detected.

The other measurement point
of the ATMEL micro controller
is connected to GND internally.

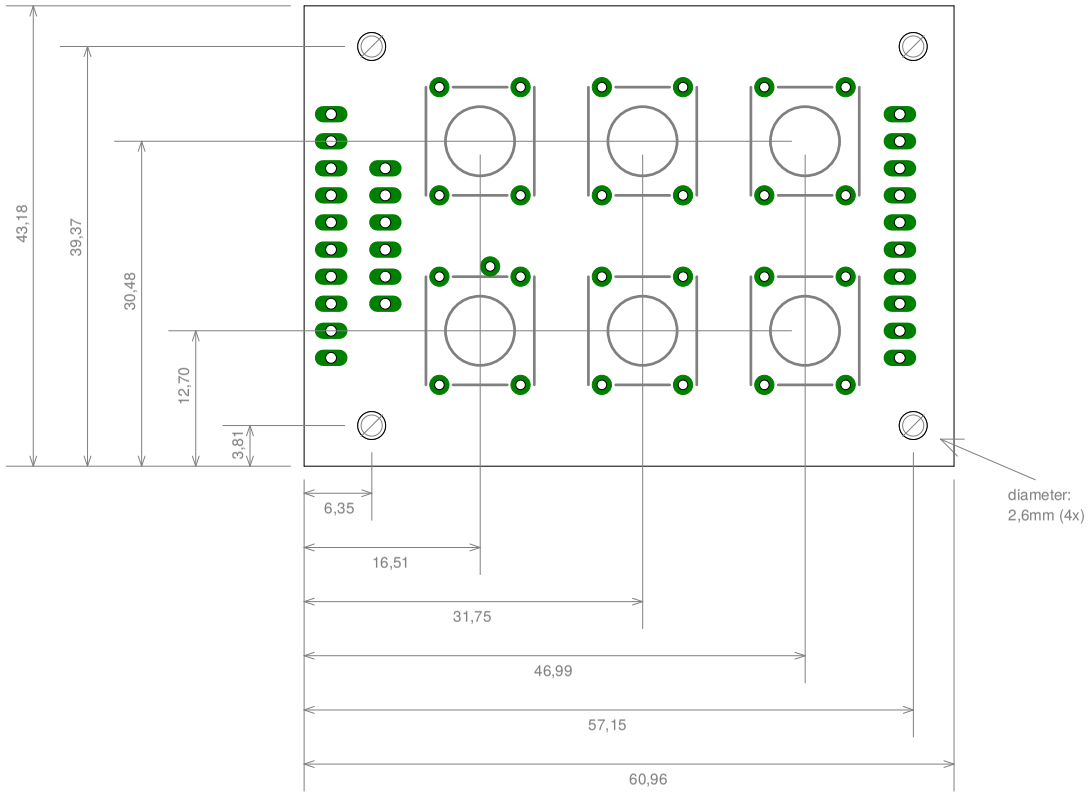
Esplora Gaming Controller	
Concept of TinkerKit switches wiring	TinkerKit-concept
	04.02.19 21:09
	Sheet: 1/1



Usage of External Digital Switches

* Don't solder RDIG_ORL and RDIG_ORR
* Solder RANA_ORL and RANA_ORR
* solder external switches at X1.1-2 and X1.5-6
* use switches OR_LEFT and OR_RIGHT as analogue inputs:
  calculate suitable values for RDN_ORL and RDN_ORR
  68k or 82k? (t.b.d.)

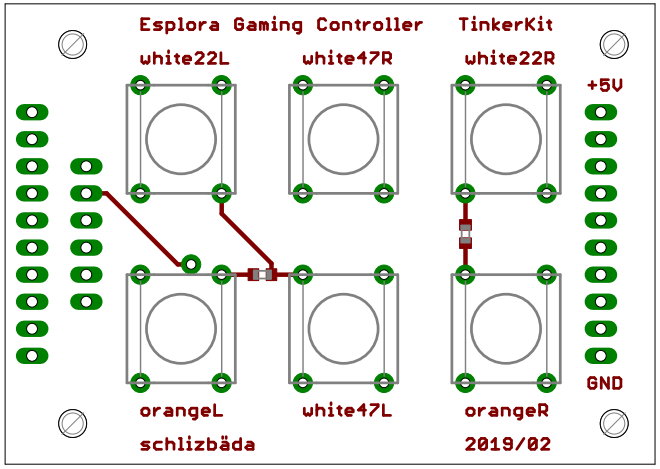
Esplora Gaming Controller -- TinkerKit buttons	
	TinkerKit-board
	03.02.19 17:04
	Sheet: 1/1



Esplora Gaming Controller -- TinkerKit buttons

TinkerKit-board

03.02.19 17:06



Esplora Gaming Controller -- TinkerKit buttons	
	TinkerKit-board
	03.02.19 17:06

Explora Gaming Controller -- TinkerKit buttons

	TinkerKit-board	03:27:20:09