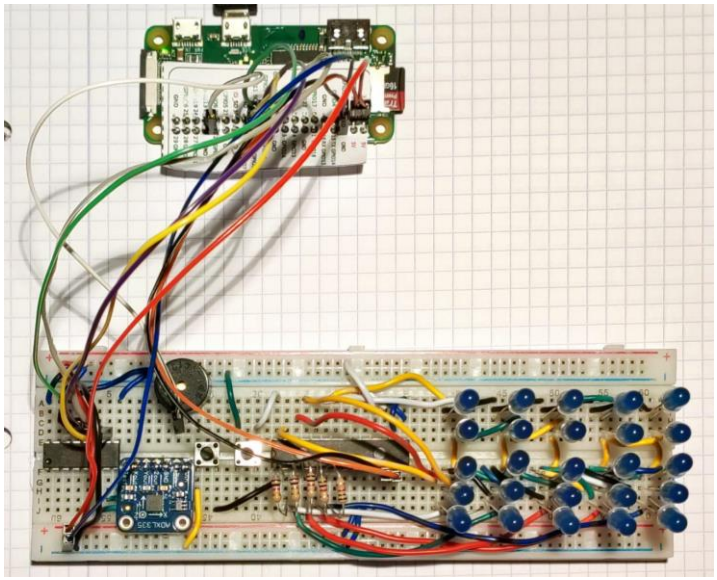

Foto 1: De schakelingen van de micro:bit

Tonen van patronen

De code die zorgt voor het tonen van patronen kan teruggevonden worden in het bestand “multiplexing-drawing-lib.rkt”. Concreet werkt dit als volgt:

Patronen worden voorgesteld als een vector van vectoren. Hiervoor wordt een matrix ADT voorzien. De implementatie van dit ADT kan teruggevonden worden in het bestand “adt-matrix.rkt”. Dit ADT bevat de volgende procedures:

Procedure	**Signatuur**	**werking**
make-matrix	$(number\ number\ any \rightarrow matrix)$	Een nieuwe matrix aanmaken
list->matrix	$(number\ number\ list \rightarrow matrix)$	Een lijst omzetten naar een matrix
matrix-ref	$(matrix\ number\ number \rightarrow any)$	Een element uit de matrix teruggeven
matrix-set!	$(matrix\ number\ number\ any \rightarrow \emptyset)$	Een element in de matrix aanpassen naar een nieuw element
matrix-rows matrix-cols	$(matrix \rightarrow number)$ $(matrix \rightarrow number)$	Aantal rijen/kolommen van de matrix teruggeven.

Tabel 1: procedures van het matrix ADT

Een matrix bevat nullen en enen. Het getal nul geeft aan dat de led in de positie van die nul uitgezet moet worden, en het getal een geeft aan dat de led in de positie van die een aangezet moet worden. In de multiplexing drawing lib kan de procedure make-pattern teruggevonden worden. Deze procedure zal, gegeven een lijst of een vector, een nieuw patroon aanmaken (door een matrix te maken met list->matrix indien het een lijst is) dat getekend kan worden met 1 van de drawing functies.

Eerst moet het multiplexing display geinitialiseerd worden. Dit gebeurt door de procedure initialize op te roepen. Deze procedure zal alle pinnen van het display op output zetten, alle kolommen uitzetten door

de pinnen van de kolommen op 0 te zetten, en alle rijen uitzetten door de pinnen van de rijen op 1 te zetten.

Patronen op het scherm tekenen gebeurt met behulp van de procedure draw. Deze procedure neemt een patroon (een matrix) en een getal, die de delay voorstelt, als invoer, en zal vervolgens een patroon tekenen op het scherm. De delay wordt doorgaans op 1ms gezet om een vast patroon te kunnen zien. Dit gebeurt door de rijen achter elkaar heel snel aan en uit te zetten waardoor een illusie van een vast patroon gecreëerd wordt wanneer draw in een loop opgeroepen wordt.

Daarnaast ondersteunt de multiplexing drawing lib extra functionaliteiten, de belangrijkste zijn:

draw-with-scrolling-effect is een procedure die een lange matrix op het scherm met een scrollend effect zal tekenen. Intern definieert deze procedure een andere procedure draw die heel gelijkaardig is met de bovenvermelde draw procedure op 1 verschil na: deze procedure zal bij het reffen in het patroon de kolom optellen met een variabele, pos. De draw procedure zal elke loop voor een bepaald aantal keren, scolling-delay, opgeroepen worden om een beetje vertraging te creëren zodat de gebruiker kan lezen wat op het scherm staat, daarna wordt de pos variabele met 1 verhoogd zodat er in het patroon met 1 kolom verschoven kan worden.

draw-number is een procedure die een getal, cijfer na cijfer, op het scherm zal tekenen. Deze procedure zal het getal eerst omzetten naar een lijst van cijfers, en zal de cijfers na elkaar op het scherm tekenen. Dit gebeurt door het cijfer te zoeken in een grote matrix, DIGITS genoemd, dat de cijfers van 0 tot en met 9 bevat, en vervolgens een stukje uit die matrix displayen dat dan het cijfer bevat.

Er wordt ook een abstractie voor het scherm gemaakt dat later gebruikt zal worden in de spellen. Dit maakt het gemakkelijker om de spellen te implementeren:

screen is een patroon dat het scherm voorstelt. turn-on! en turn-off! zijn twee procedures die zorgen voor het aan-/uitzetten van een led in het scherm gegeven de coördinaten van die led. draw-screen is een procedure die het scherm zal tekenen. Deze procedure moet in een lus opgeroepen worden.

Aansturen van het “sterren vangen”-spel d.m.v. de accelerometer

Eerst moet de accelerometer gekalibreerd worden. Dit gebeurt door de procedure calibrate-loop op te roepen. Luisteren naar de input van de accelerometer gebeurt m.b.v. de procedure read-axis.

In de procedure listen-to-input zal geluisterd worden naar de input van de accelerometer voor de x-as en de y-as, en de waarden die daaruit komen zullen herschaald worden naar een waarde tussen 0 en 10. In listen-to-input zit een conditional die bepaalt wat er moet gebeuren bij het kantelen van de accelerometer rond de x-as of de y-as. Om code-duplicatie te vermijden wordt de procedure axis-rotated gedefinieerd. Deze procedure zal de vorige positie van de speler uitzetten en de nieuwe positie aanzetten. Listen-to-input neemt een procedure-object als argument, namelijk de actie die moet gebeuren wanneer de accelerometer rond een bepaalde as geroteerd wordt, wanneer er rond de x-as geroteerd wordt, moet de ”y-coördinaat” (de kolom) van de positie van de speler bijvoorbeeld verhoogd of verlaagd worden. Een variabele rotated-to-left? wordt geïntroduceerd om code-duplicatie te vermijden. Deze variabele wordt op true gezet wanneer er naar links gekanteld wordt, en dan moet de y-coördinaat verlaagd worden, anders moet die verhoogd worden. Hetzelfde idee gebeurt langs de y-as.

Cruciale code

Aangezien dat delays ervoor zorgen dat het scherm flikkert, worden ze zo veel mogelijk vermeden. Daarom wordt gebruikgemaakt van timers. Een timer is een object (een ADT) dat aangeeft of een bepaalde tijdsperiode al gepasseerd is. Er worden twee soorten timers geïmplementeerd: Een gewone timer en een timer met een schakelaar. Deze timers begrijpen de volgende procedures:

Procedure	**signatuur**	**werking**
make-timer make-timer-with-a-switch	$(number\ boolean \rightarrow timer)$	Gegeven een getal, de duur van de timer, en een boolean, als die boolean true is, is de timer in milliseconden, anders in seconden, wordt een timer aangemaakt.
time-passed?	$(\emptyset \rightarrow \emptyset)$	Een predikaat dat slaagt als de tijd voorbij is. Deze procedure zal de timer ook resetten.
reset-time!	$(\emptyset \rightarrow \emptyset)$	De timer resetten.
Enkel voor de timer met een schakeling:		
Start-the-count!	$(\emptyset \rightarrow \emptyset)$	De timer aanzetten.

Tabel 2: de procedures van de timer ADTs

Aan de hand daarvan worden een paar vaak voorkomende timers gedefinieerd, zoals een timer voor 100 milliseconden, 200 milliseconden, 5 seconden en 1 minuut. Timers worden gebruikt als alternatief voor delays in de oefeningen, bijvoorbeeld om de knoppen voor 100 milliseconden uit te zetten. Er wordt dan een variabele geïntroduceerd die aangeeft of de input uitgezet moet worden, en die variabele wordt op true gezet wanneer er input is gebeurd, en na zoveel tijd terug op false. Timers worden in OOP-stijl geïmplementeerd zodat elk object zijn eigen lokale toestand heeft apart van de andere timerobjecten.

Piezo-Player

Voor de vijfde oefening heb ik een programma gemaakt dat muziekstukken afspeelt. Op het scherm verschijnen er patronen die muziekstukken voorstellen, de gebruiker kan doorheen deze patronen wandelen door op de linkerknop te drukken, en een patroon selecteren en het muziekstuk afspelen door op de rechterknop te drukken. Het programma te beëindigen kan door op beide knoppen tegelijkertijd te drukken wanneer er geen muziekstuk aan het afspelen is. De muziekstukken zitten in het bestand songs.rkt. De procedure make-song zet de muziekstukken om naar een associatielijst waarbij in de car de frequentie zit en in de cdr de duur van de frequentie in seconden. Elk muziekstuk komt later ook in een lijst te zitten met het muziekstuk zelf in de car en het tempo van de muziek in de cadr. Alle muziekstukken worden in een vector geplaatst om het zoekwerk te vereenvoudigen (vector-reffen in de vector).

In het bestand van de piezo-player (oefening-5.rkt) zitten er twee procedures: buzz en play. Buzz zal 1 frequentie afspelen voor een bepaalde duur, en play zal een associatielijst van frequenties afspelen voor een bepaalde duur op een bepaald tempo. In de procedure next-pos! zit een if test om na te gaan of we op het einde van de patronen zitten, als dat zo is, keren we terug naar het begin. De patronen zitten in een vector in het bestand “patterns.rkt” en de plaats van elk patroon in de vector komt overeen met de plaats van het overeenkomstige muziekstuk in muziekvector. Op die manier wordt een eenvoudige dictionary aangemaakt en kunnen muziekstukken en patronen gemakkelijk opgevraagd worden.