Assignment 2: Digital Pet

Create a lovable digital pet on your microbit

Photo by COSMOH LOVE on Unsplash

On this page

Digital pets like the Tamagotchi or Nintendogs are computer simulations of a friendly animal companion created as a toy or video game. These toys feature an animated depiction of the animal and inputs through button that allow the pet's human owner to care for it

inputs through buttons that allow the pet's human owner to care for it or interct with it. Usually the owner gets to play with it, but also needs to provide food or entertainment in order for the pet to be happy[1].

Your task in this assignment is to create a *digital pet* in your microbit!

You are going to write an ARM assembly program that simulates a digital pet. It should **display the pet** on the LED display, and allow a human to **interact** with the pet using the microbit's buttons. Your digital pet should provide companionship (that is, it's **engaging** to look at) but also responsibility (that is, it **requires** interaction).

Your pet should have some kind of *state* (e.g., it's levels of health and happiness) that is stored in memory, and you will need to use *interrupts* to receive input from the microbit's buttons.

Outline

- **Deadline:** 2022-05-20 23:59 Canberra time.
- **Assignment template:** link
- **Specification:** keep reading :-)

Specification

Here's a technical specificiation for your assignment.

Your program:

- **must** be written in **ARMv7 assembly** using the assignment template (link)
- **must** use the LED display to show a digital pet
- **must** use (at least one) data structure in memory to store the "state" of the digital pet
- **must** use interrupts to detect interactions with the microbit's buttons
- **must** work when the microbit is powered over USB but not connected to a computer (that is, it works after you upload it and plug into a USB charger)
- **must** be *engaging* and *require* interaction for a fun experience over 1-3 minutes
- *can* use the speaker to create sound
- *can* use other inputs (e.g., microphone, IMU)
- *can* use **any** peripheral available on the microbit

Finally, you program **must** be accompanied by a design document (maximum: **600 words**). The design document must explain:

- what your design is (and how it meets the assignment specification)
- how you accomplished it

- why your design choices were appropriate for the task

You **must** follow the assignment specification for this project (both **shoulds** and **should nots**). Our expectation is with the above limitations every student will come up with a unique digital pet.

Rules and Policies

- this is an individual assessment task
- late submission is not permitted without an extension

Marking Criteria

Your assignment will be evaluated on the following criteria (total 20 marks):

1. Sophistication of your design and how it meets the assignment specification. (20%, 4 marks)
2. Sophistication of your implementation in ARM-v7 assembly language. (50%, 10 marks)
3. Sophistication of analysis and evaluation of why your implementation is correct and appropriate for your design and what limitations it might have. (20%, 4 marks)
4. Sophistication of communication and expression. (10%, 2 marks)

Item 2 will be evaluated primarily through your program code. Items 1, 3, and 4 will be evaluated through your design document.

Submission

Submission is through GitLab, the most recently pushed commit of your fork of the assignment template before the deadline is taken to be your assignment submission.

Getting Started

1. read this assignment page **completely**
2. **presubmission task:** fork and clone the assignment template, complete your `design-proposal.md` (commit and push it!) then participate in the presubmission lab
3. think about what kind of data structure would be required for a digital pet (see lab 7 for help).
4. complete labs 8 and 9 and think about how that work will help you to write a more complex interactive program
5. think about your first assignment and consider how you can create a useful and engaging display for your digital pet.

6. after you create a basic prototype, ask a friend or family member to try it to see if the interaction is intuitive and fun
7. make a mistake or get stuck, then ask a good question on the course forum.

Ideas for Extensions

So you've made a basic digital pet and now you want to take it to the next level? Here's some ideas for making a sophisticated idea:

- improve the LED display to make it really stand out, e.g., by using PWM and smooth animations, catchy visuals and interesting effects, etc.
- write a program that creates pets randomly and provides variation every time you play (e.g., a *generative* program)
- implement a random number generator and use it in your program
- use the speaker to create an interesting synthesised sound to go with your digital pet.
- use some kind of network connectivity (LEDs to light sensor, bluetooth, UART, etc) to send your pet to another microbit.
- have your pet react to changes in environment (use the motion sensor or light sensors).
- use non-volatile memory to store your digital pet on the microbit even when the power is disconnected.

These are only a couple of extension ideas, if you have any other great ideas feel free to chat with us on the course forum course forum.

Checklist

- you have tested that your code works when "uploaded" to a microbit and when the microbit is plugged into a USB charger or power bank
- you have checked with at least one other person that your digital pet is engaging (show your family! good chance to show off what you're learning at uni!)
- you have committed and pushed your program code to your own fork of the assignment template — **check this on the gitlab website!**
- you have filled out, committed, and pushed `design-document.md` — **check this on the gitlab website!**
- you have filled out, committed, and pushed your statement of originality. — **check this on the gitlab website!**

FAQ

Do I have to write a design document?

Yes! 50% of the marks for this assignment are evaluated through the design document. If you don't write one you will get (almost) zero for that half of the assignment.

How do I write a design document?

Have a look at the design document page for advice.

Make sure you are answering the questions in the specification and stay within the word limit.

Writing a clear and concise document is a challenge, but we believe in you.

How am I supposed to display a digital pet in 25 LEDs?

Having completed assignment 1 you should know what the microbit display is capable of (not much in terms of resolution).

We suggest thinking *small* in terms of what you display, we have seen some compelling characters in the assignment 1 solutions from 1-pixel entities to simple 5x5 sprites. It is possible to do a lot with a little and to add personality and charm on a limited screen!

How long should I expect the tutors to play with my digital pet?

Expect tutors to spend a couple of minutes (1-3) with your pet, don't write a program that **requires** an hour or a day of interaction to make sense.

The original tamagotchi was also only fun for a few minutes, its just that it would also be fun to repeat that experience regularly throughout a day as the pet grew and responded to your inputs.

Of course, you can create a program that has longer-term features if you want (and write about it in your design document), but make sure it still makes sense in the 1-3 minutes timeframe. If you want to do this, maybe you could include some way to "skip time" so your tutors can explore all the features you have created. If you write a truly fun program, your tutor might get obsessed with your pet and skip labs to care for it! We can't decide if this is good or bad.

I'm a zoomer and I don't know what a digital pet is. Help!

If you don't know what a "tamagotchi" or another kind of digital pet is here's some videos explaining it:

- Tamagotchi gameplay
- Tamagotchi unboxing and setup

Tamagotchis were litle plastic toys (chunky keychain size) from the 90s with a basic display, three buttons and a little beepy speaker. The idea was that they had a little "pet"

program that you had to care for and play with. Over time, as you played with your pet, it would gain stats (skills, health etc), and allow you to interact in more interesting ways.

Eventually your pet would grow old (oh no..) and die (omg!). Kids were obsessed with them (waaaay before smartphones and tablets).

Tamagotchis had a higher res display than the microbit, but the microbit is a *much* more advanced computer, so a microbit-based digital pet *could* be a lot better than a tamagotchi if you can find a clever way to display your digital pet with the basic LED screen.

Your tutors will have a couple of Tamagotchis on hand in labs during weeks 8, 9, and 10 to help get some digital pet inspiration.

What do you mean by "engaging"?

We mean that your digital pet should be interesting enough to play for a few minutes. It should **require** the player to interact with it (using some input method on the microbit) in order to see changes occur in the pet or to look after it.

It's a digital pet, does it have to be an animal (e.g., like a rabbit)?

No, it can be anything you like that requires "care" in some way and is entertaining to play with. Kids sometimes have pet rocks (not sure why) so if you can think of a way to make that engaging, go for it.

I've done X, is that good enough to get Y marks?

We don't give out marks before assignments are completed, but it's fair to wonder "How hard should I work on this?"

Realistically there are ways to complete this assignment with minor modification of what you have done in labs. Those types of submissions will be likely to get in the range of 50-59 indicating "satisfactory" performance, but not what we would call "good".

Putting in a normal amount of further effort to create a submission that is "good" or "very good" will result in a mark between 60-79. Most students will be in this category. Don't neglect your design document as **explaining** your program is worth the same number of marks as the program itself.

Getting higher than 79 requires an "excellent" submission across all marking criteria. This requires sophisticated engagement with the task and probably some kind of extension.

For very high marks (90+): "we know it when we see it". Only a few students will be in this category and they will have definitely put in a lot of effort to create a sophisticated extension on the basics. They will also have written an impeccable design document.

At the opposite end of the scale, if we see a submission that doesn't meet the specification

(e.g, it doesn't use the inputs, or doesn't store data about the digital pet in memory), then the mark will be below 50. Hopefully, there are only a few students in this category.

Sorry that we can't be more specific than that.

My program doesn't work, can I email you for help?

Sorry, you won't get help over email or Teams. We provide a course forum which is the **only** way we are able to help.

Forum posts related to your assignment submission **must** be "private to instructors" (as for any individual assessment task).

Do I have to use memory? It seems hard.

Yes, storing a data structure in memory is a requirement for this assignment.

We've seen a lot of students avoid using memory because it's slightly more complicated than using registers.

In the context of the course, this is a big mistake. Practicing how to `str` and `ldr` in assignment 1 will make a big difference in your ability to complete the later tasks in the course.

Realistically, there's a very limited number of registers on the microbit and many tasks are made easier by using memory at the appropriate time.

It's [5 minutes, 60 minutes, 12 hours] before the deadline and my CI Jobs aren't finishing!

Unfortunately on the day that an assignment is due, when many students are pushing updates at once, the CI servers can't keep up. You may not see your CI jobs finish before the deadline. You will just have to manually check that your files have been submitted correctly.

If there's any issues with your git repository *after the deadline*. Please let us know (after the deadline) through a private piazza post and there may be something we can do.

How do I know my assignment has been submitted?

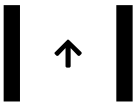
If:

1. the files in *your fork* of the assignment are correct (i.e., the files you intend to submit) when checking on the gitlab website
2. the time is before the deadline

then your assignment has been submitted (well done!).

Please don't ask us to "check", we would be just doing exactly the same thing as the above steps which you can do yourself.

1. When your course convenor was a kid, digital pets became **so popular** with kids at primary school that they were banned from bringing them to class. This was bad because if you didn't interact with these pets, they **died!** Pretty tough lesson in life for a 10 year old. ↩



Acknowledgement of Country

The Australian National University acknowledges, celebrates and pays our respects to the Ngunnawal and Ngambri people of the Canberra region and to all First Nations Australians on whose traditional lands we meet and work, and whose cultures are among the oldest continuing cultures in human history.

Updated: 15 May 2022

Responsible Officer: School of Computing Director

Page Contact: COMP2300 Course Convenor