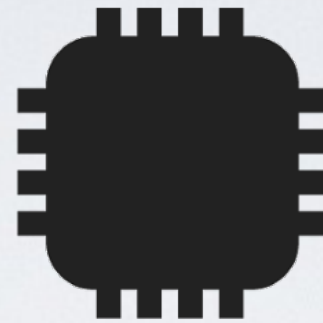




Kart Programming

Pierre Roduit | François Corthay
Christopher Metrailler | Oliver Gubler | Michael Clausen

Goals

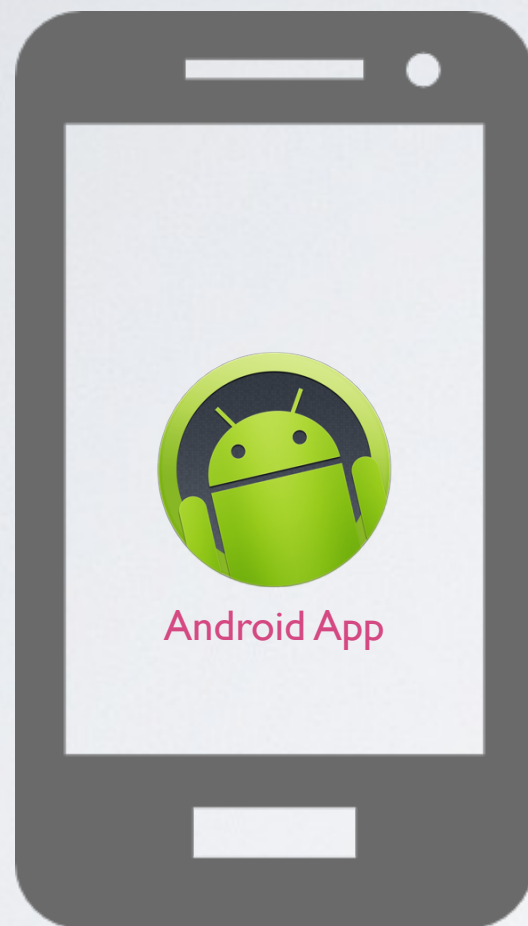


Use basic elements seen in ELN course.



Use basic elements seen in INF I course.

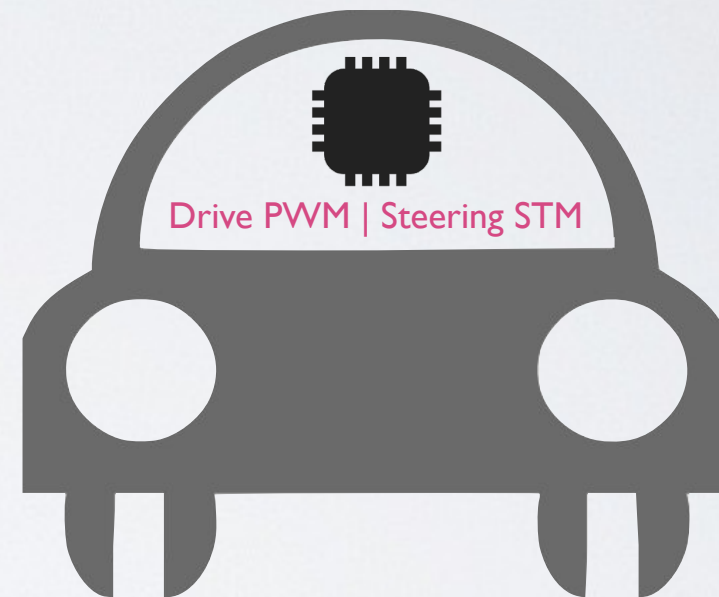
Goals



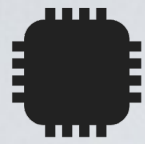
Galaxy Nexus



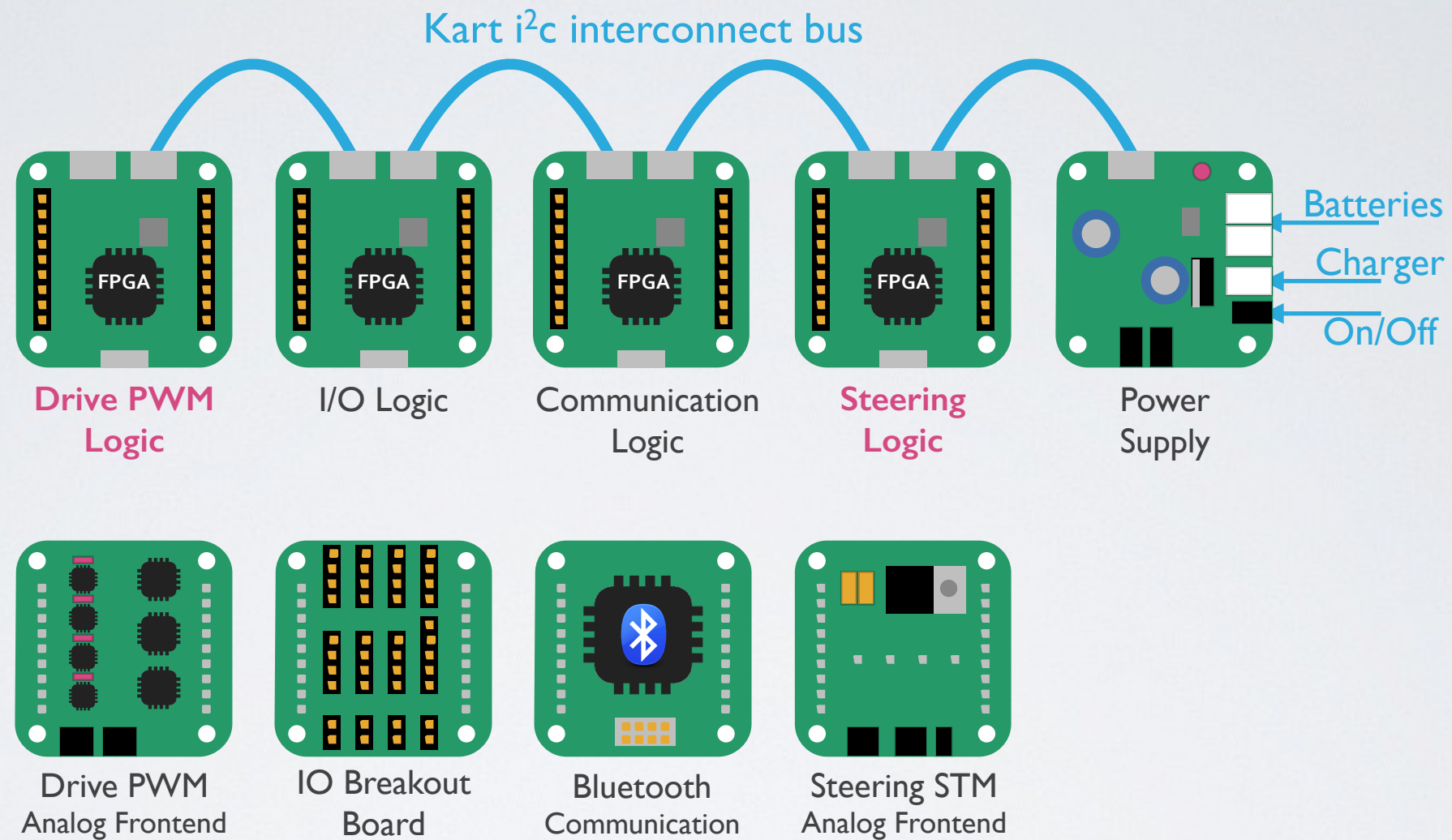
Bluetooth

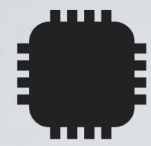


Kart

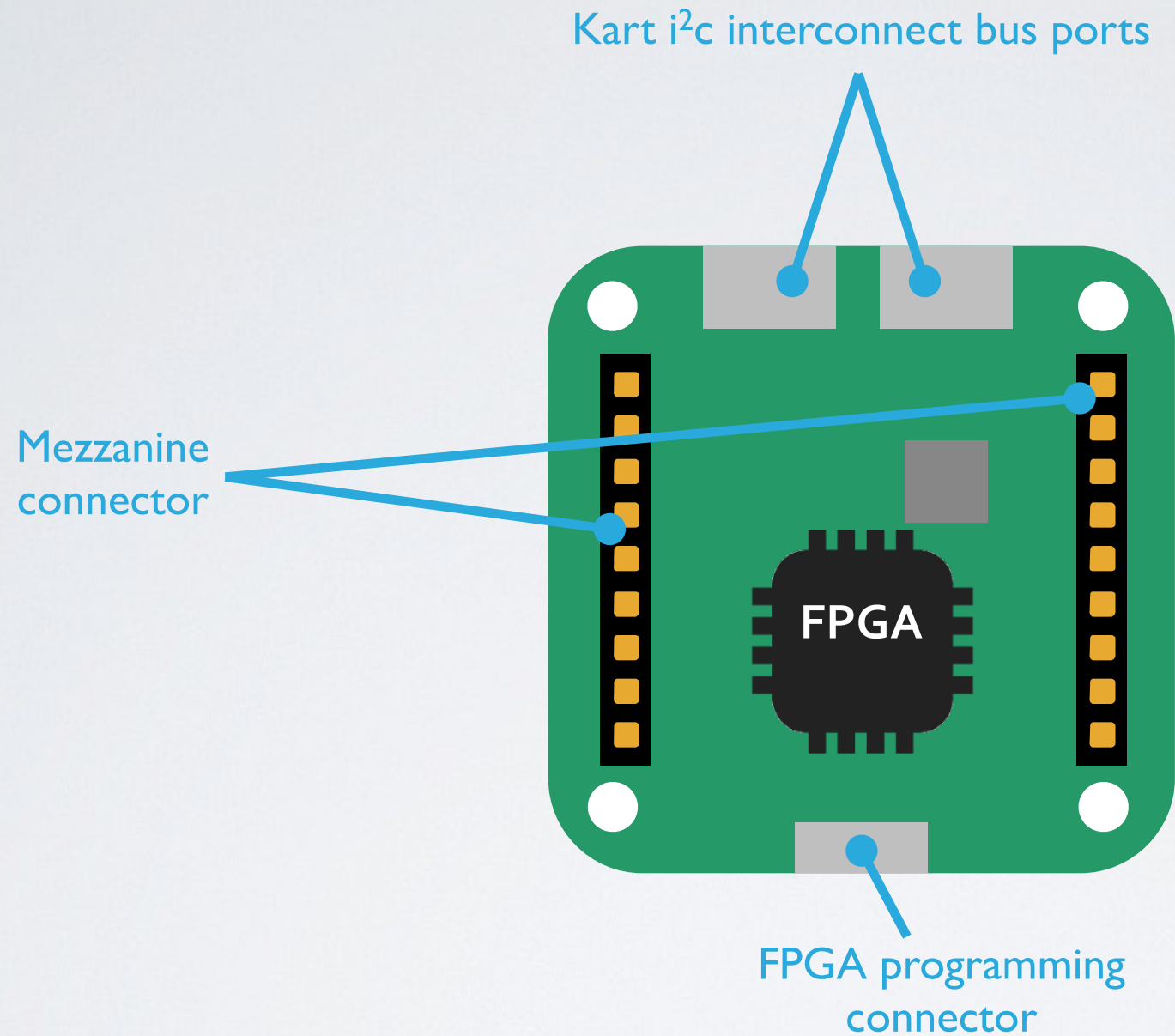


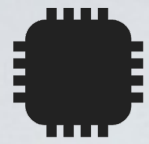
Modular concept





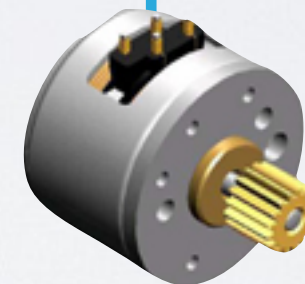
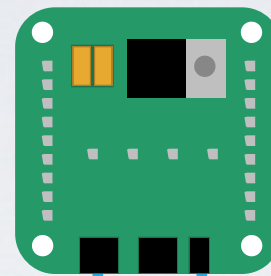
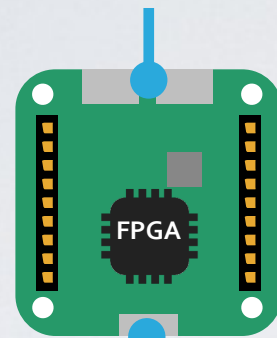
Generic FPGA Board





Stepper Motor

Kart i²c interconnect bus

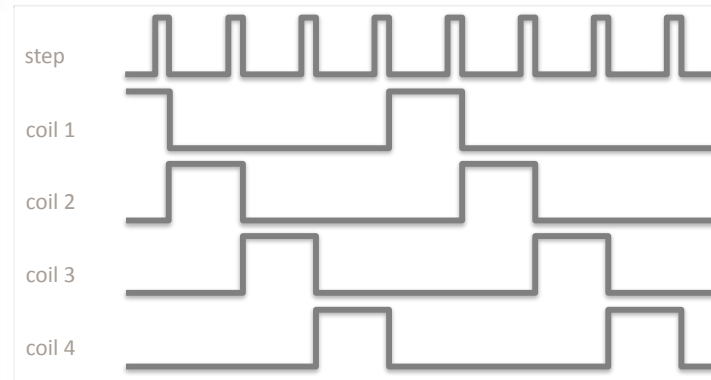


Limit switch

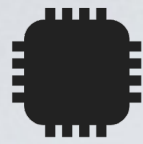
Stepper motor

```
1  -- Company: Young Embedded Systems LLC
2  -- Engineer: Gene Breneman
3  --
4  -- Module Name:  ARM_SEQ_RAM - Behavioral
5  -- Revisions:
6  -- 0.01 - 07/02/2007 File Created
7  -- Additional Comments:
8  --
9  --
10 library IEEE;
11 use IEEE.STD_LOGIC_1164.ALL;
12 use IEEE.STD_LOGIC_ARITH.ALL;
13 use IEEE.STD_LOGIC_UNSIGNED.ALL;
14
15 entity ClkDiv is
16   Port ( InByte : in STD_LOGIC_VECTOR(3 downto 0);    --<-- Seq_CPLD
17         RegSel : in STD_LOGIC_VECTOR(1 downto 0);    --<-- Seq_CPLD
18         RegStb : in STD_LOGIC;                       --<-- Seq_CPLD
19         Mclk : in STD_LOGIC;                         --<-- OSC
20         SeqReset : in STD_LOGIC;                     --<-- Power Monitor
21         ADC_Clk : out STD_LOGIC);                   -->-- ADC
22 end ClkDiv;
23
24 architecture Behavioral of ClkDiv is
25   signal ADC_div : STD_LOGIC_VECTOR(5 downto 0) := "001111";
26   signal ADCClk : STD_LOGIC := '0';
27   signal ClkSel : STD_LOGIC_VECTOR(2 downto 0) := "100";
28
29 begin
30
31   ClkDivP : process(Mclk,SeqReset)
32   begin
33     if SeqReset = '0' then
34       ADCClk <= '0';
35       ADC_div <= "001001";
36     elsif Mclk = '0' and Mclk'event then
37       if ADC_div = "000000" then
38         ADCClk <= not(ADCClk);
39         case ClkSel is
40           when "000" => -- 20MHz - divide by 2
41             ADC_div <= "000001";
42           when "001" => -- 10MHz
43             ADC_div <= "000001";
44           when "010" => -- divide by 4
45             ADC_div <= "000100";
46           when "011" => -- divide by 10
47             ADC_div <= "001001";
48           when "100" => -- 2MHz
49             ADC_div <= "001001";
50           when "101" => -- divide by 20
51             ADC_div <= "001001";
52           when "110" => -- 1MHz
53             ADC_div <= "001001";
54           when "111" => -- divide by 40
55             ADC_div <= "001001";
56           when others => -- 400KHz
57             ADC_div <= "001001";
58         end case;
59       end if;
60     end if;
61   end process;
62 end Behavioral;
```

Stepper motor control

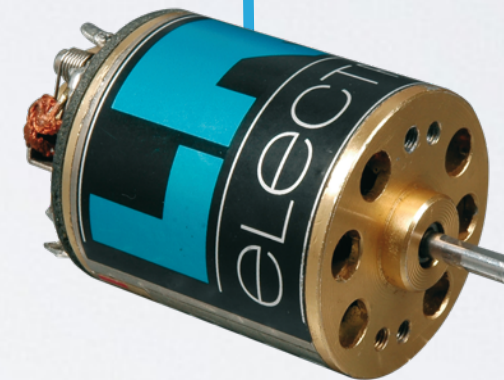
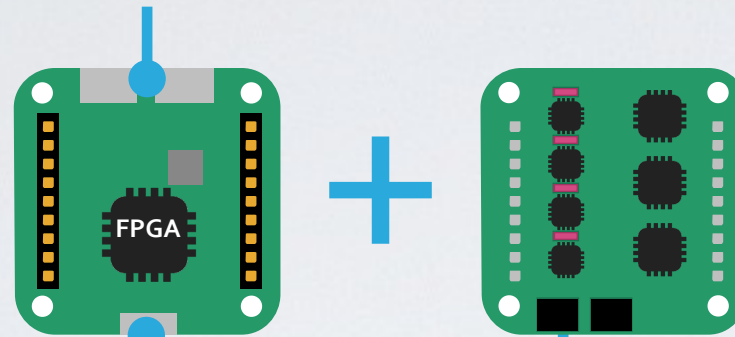


Coil sequence



Drive Motor

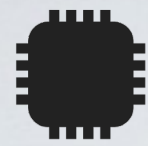
Kart i²c interconnect bus



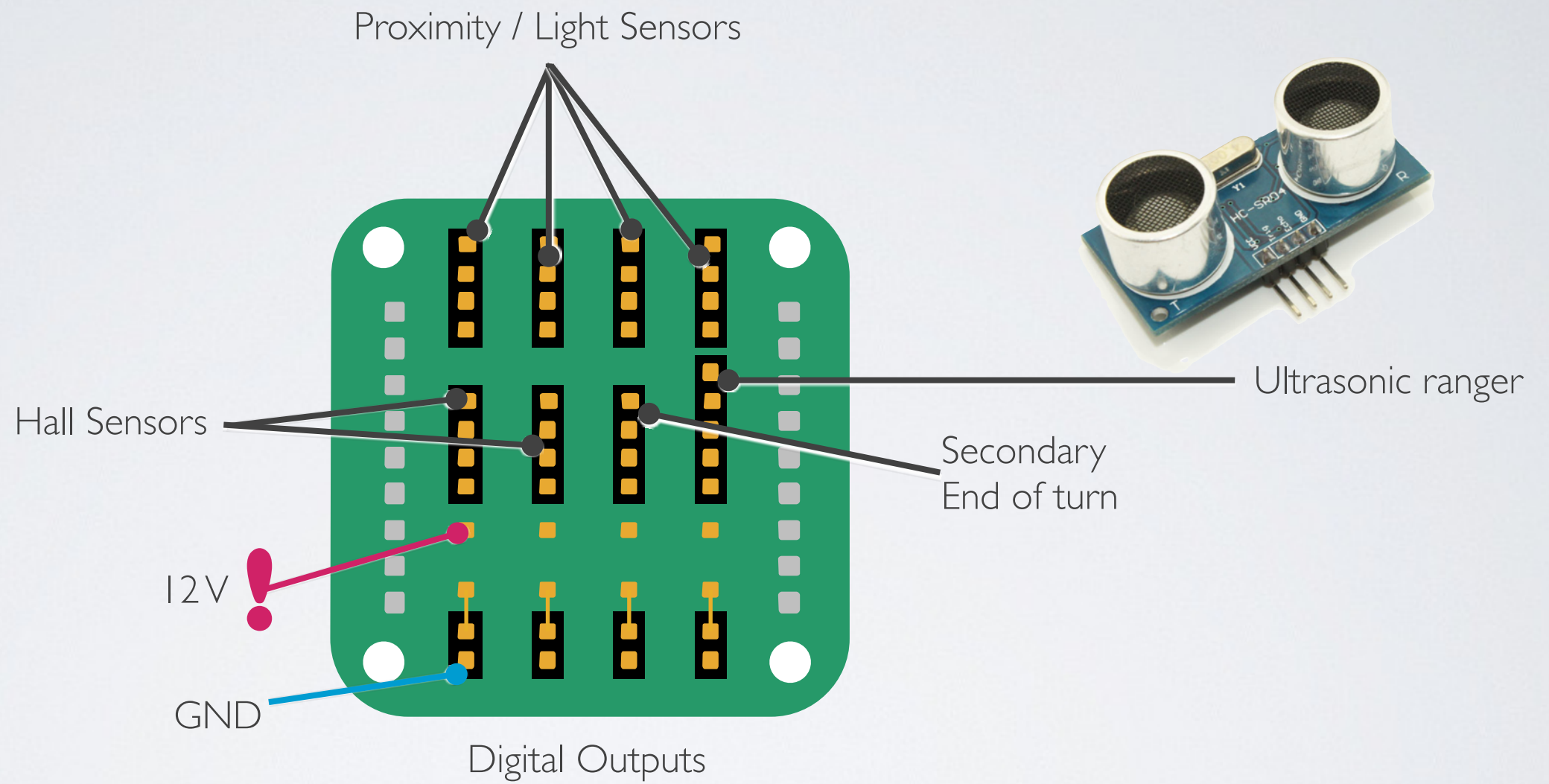
Drive motor

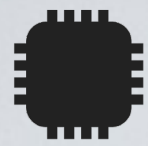
```
1
2 -- Company: Young Embedded Systems LLC
3 -- Engineer: Gene Breniman
4 -- Module Name:  ARM_SEQ_RAM- - Behavioral
5 -- Revisions:
6 -- 0.01 - 07/02/2007 File Created
7 -- Additional Comments:
8
9
10 library IEEE;
11 use IEEE.STD_LOGIC_1164.ALL;
12 use IEEE.STD_LOGIC_ARITH.ALL;
13 use IEEE.STD_LOGIC_UNSIGNED.ALL;
14
15 entity ClkDiv is
16   Port ( InByte : in STD_LOGIC_VECTOR(3 downto 0); --<-- Seq_CPLD
17         RegSel : in STD_LOGIC_VECTOR(1 downto 0); --<-- Seq_CPLD
18         RegStrb : in STD_LOGIC; --<-- Seq_CPLD
19         MClk : in STD_LOGIC; --<-- OSC
20         SeqReset : in STD_LOGIC; --<-- Power Monitor
21         ADC_Clk : out STD_LOGIC; -->-- ADC
22   end ClkDiv;
23
24 architecture Behavioral of ClkDiv is
25   signal ADC_div : STD_LOGIC_VECTOR(5 downto 0) := "001111";
26   signal ADCClk : STD_LOGIC := '0';
27   signal ClkSel : STD_LOGIC_VECTOR(2 downto 0) := "100";
28
29 begin
30
31   ClkDivP : process(MClk,SeqReset)
32   begin
33     if SeqReset = '0' then
34       ADCClk <= '0';
35       ADC_div <= "001001";
36     elsif MClk = '0' and MClk'event then
37       if ADC_div = "000000" then
38         ADCClk <= not(ADCClk);
39         case ClkSel is
40           when "000" => -- 20MHz - divide by 2
41             ADC_div <= "000001"; -- divide by 4
42           when "001" => -- 10MHz
43             ADC_div <= "000100"; -- divide by 10
44           when "010" => -- 4MHz
45             ADC_div <= "000100"; -- divide by 10
46           when "011" => -- 2MHz
47             ADC_div <= "001001"; -- divide by 20
48           when "100" => -- 1MHz
49             ADC_div <= "001001"; -- divide by 40
50           when others =>
51             ADC_div <= "001001"; -- 400KHz
52         end case;
53       end if;
54     end if;
55   end process;
56 end ClkDiv;
```

Drive PWM control



IO Board





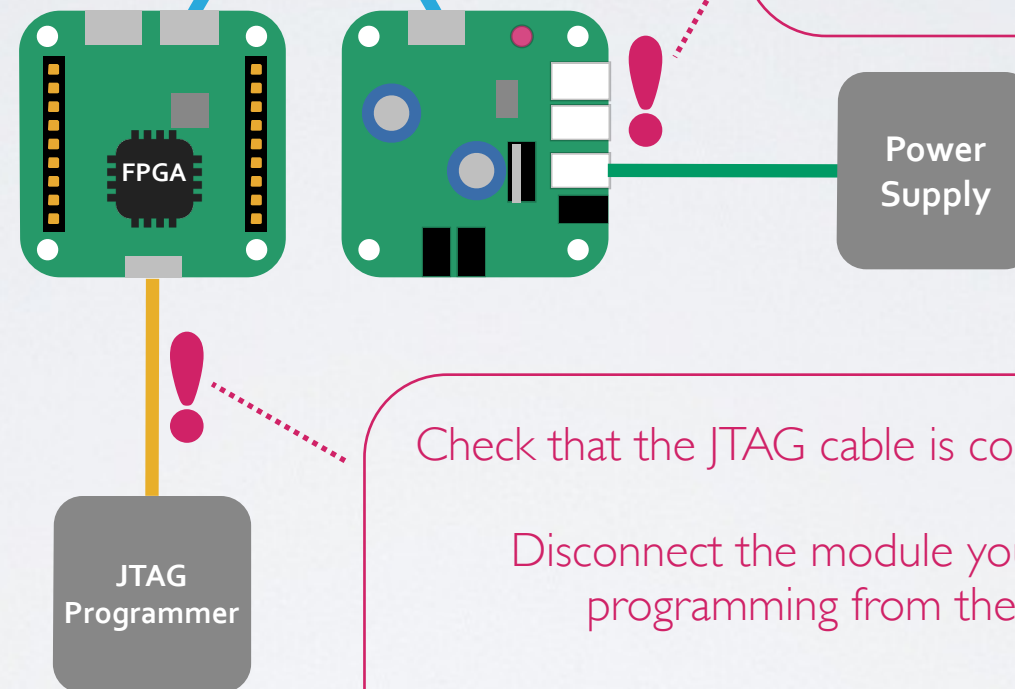
Avoid Hardware Damages

Double-check that the cable/connector are connected correctly.

Never connect JTAG connector cable here.

Power the circuit using a laboratory power supply.

The batteries should never be connected during development!



Check that the JTAG cable is connected correctly!

Disconnect the module you are actually programming from the I2C bus!

Never connect a I2C bus cable to this connector!

If you connect something wrong, the FPGA might be **damaged**.

The costs to change a FPGA are about **50 SFr**.

You will be charged for the reparation if you did not follow this guidelines!

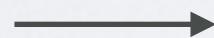
- **Control block for DC motor**
 - Pulse Width Modulation (PWM) generator
- **Control block for stepper motor**
 - 4 Coil forward/backward sequence generator
- **Various sensors and actuators**
- **Anti-collision emergency stop**
 - IR distance sensors



Presentation of blocks and simulation results during morning of the last day

All mandatory features

Direction Stepper control
Speed PWM control
Hall sensor counter



4.0

+

1

=

Grade

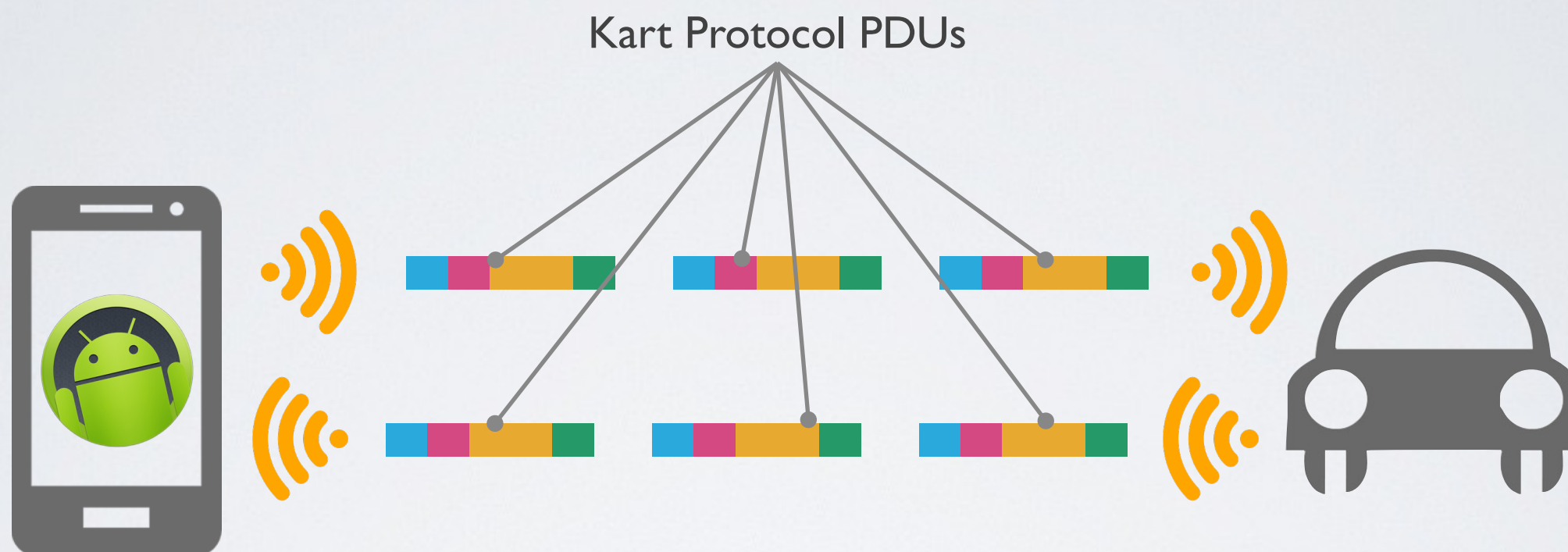
Per optional feature

Ultrasound sensor
Emergency Stop (Proximity sensor)
Other improvements



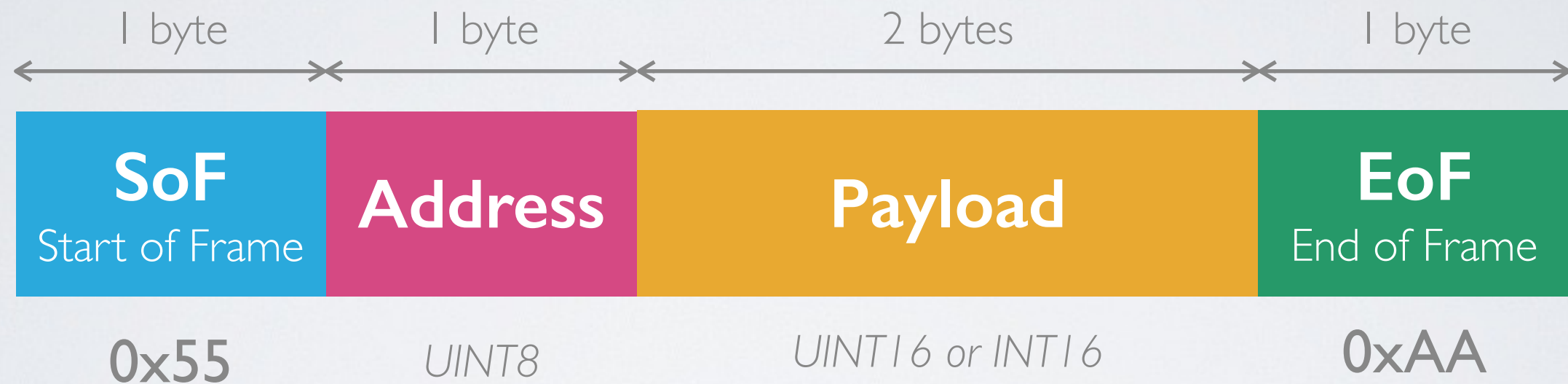


Remote Control Protocol





Remote Control Protocol PDU





Remote Control Protocol Addresses



Address	Type	Description
0x00	UINT16	Drive motor PWM period.
0x01	INT5	Drive motor speed [-15..15] (negative=backwards).
0x02	UINT16	Steering motor step period (speed proportional to 1/period).
0x03	UINT16	Steering position set point.
0x04	UINT5	Steering end switch address.
0x05	UINT5	Hardware settings (Enumeration mask).
0x06	UINT4	LED control.
0x15	UINT16	Update interval (from kart to phone) in ms.



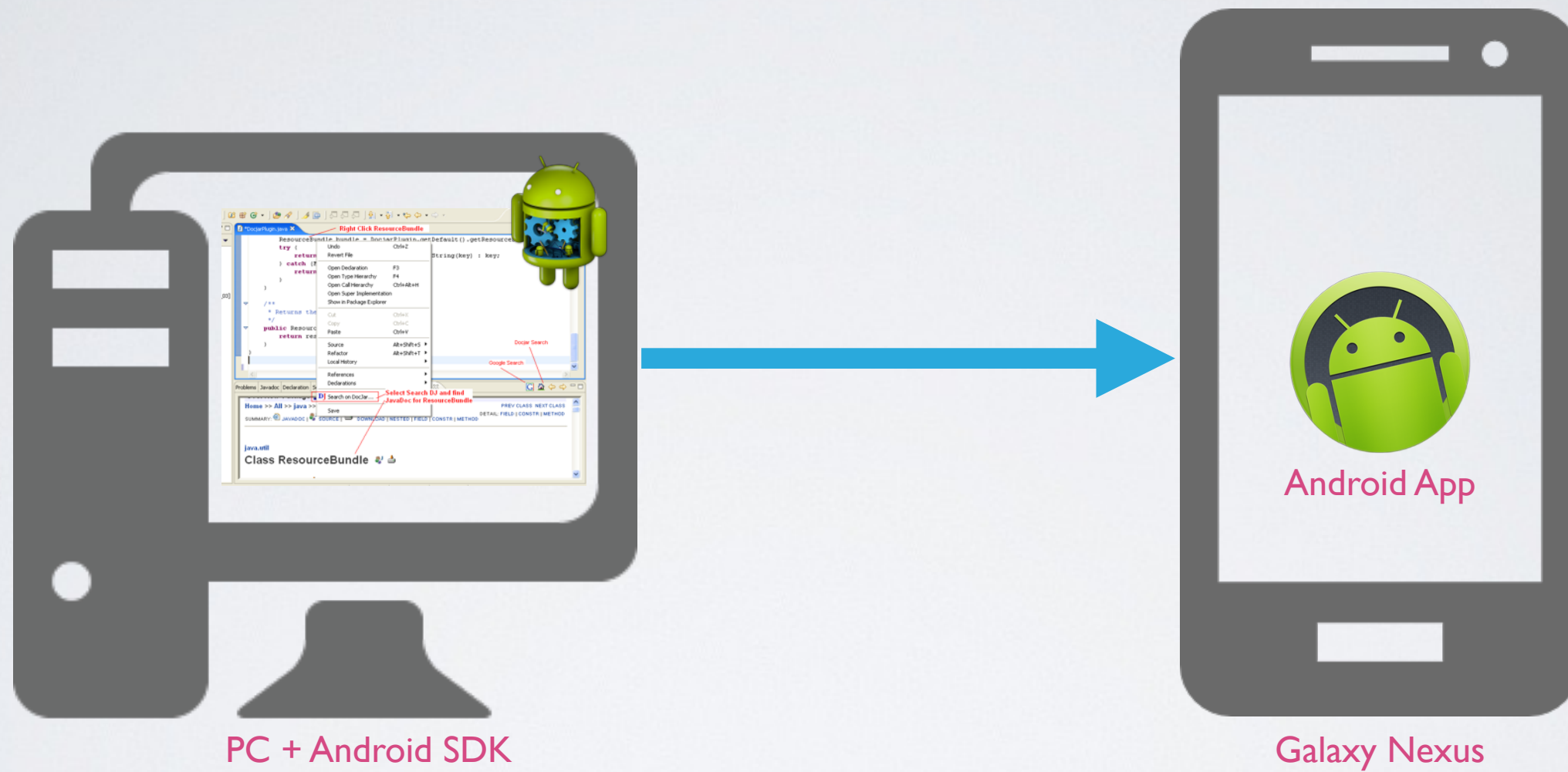
Remote Control Protocol Addresses



Address	Type	Description
0x00	UINT16	Hall sensor 1 speed count.
0x01	UINT16	Hall sensor 2 speed count.
0x02	UINT1	Steering angle reached (1=reached, 0=busy).
0x03	UINT16	Actual steering position.
0x04	UINT1	Steering end contact state (0=contact closed).
0x05	UINT16	ADC value of battery voltage level.
0x06	UINT16	Distance (Ultrasonic sensor)
0x08	UINT16	Proximity 1 (IR sensor)
0x09	UINT16	Proximity 2 (IR sensor)
0x0A	UINT16	Proximity 3 (IR sensor)
0x0B	UINT16	Proximity 4 (IR sensor)
0x0C	UINT16	Ambient Light 1 (IR sensor)
0x0D	UINT16	Ambient Light 2 (IR sensor)
0x0E	UINT16	Ambient Light 3 (IR sensor)
0c0F	UINT16	Ambient Light 4 (IR sensor)

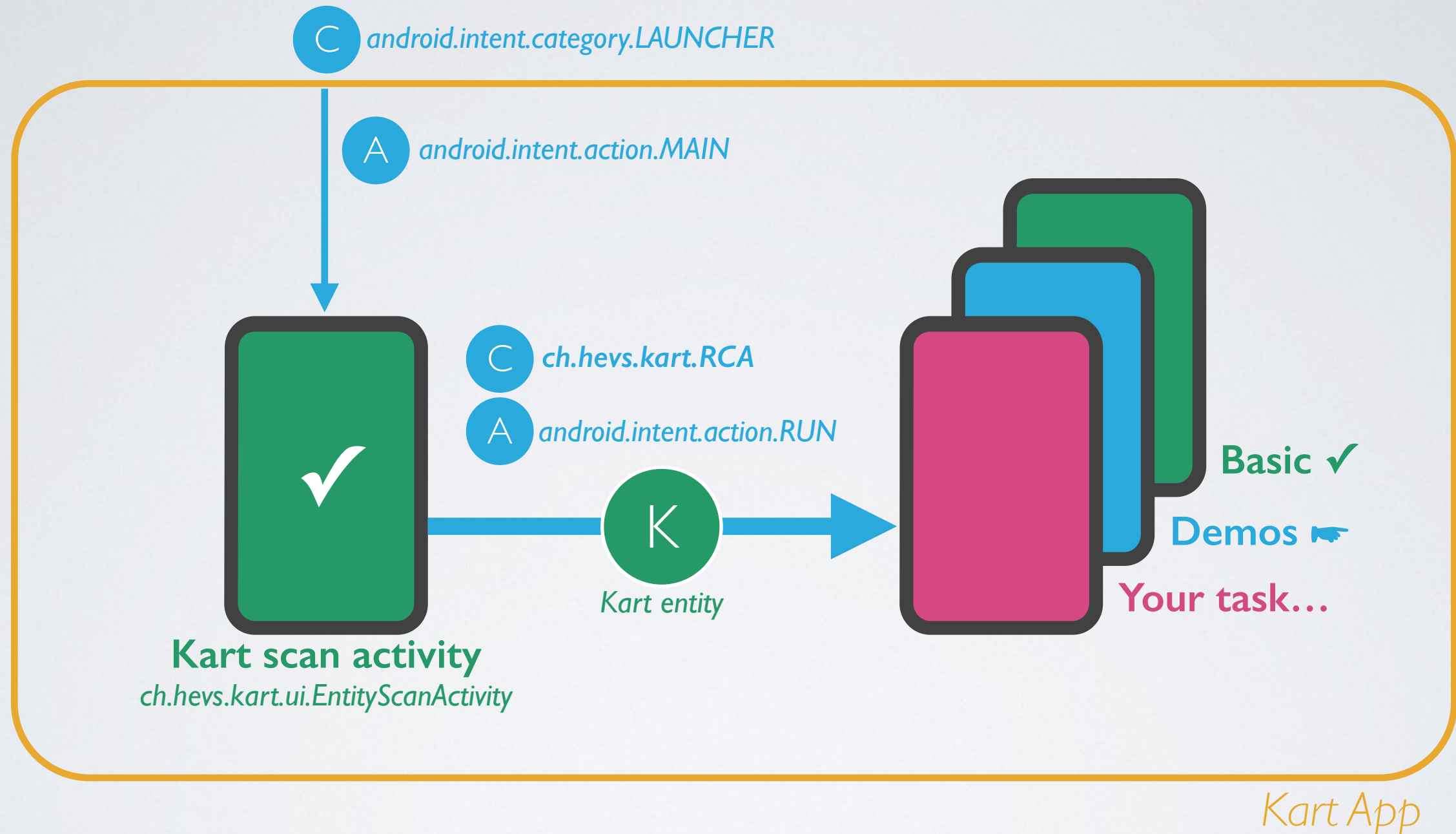


Remote Control Android App



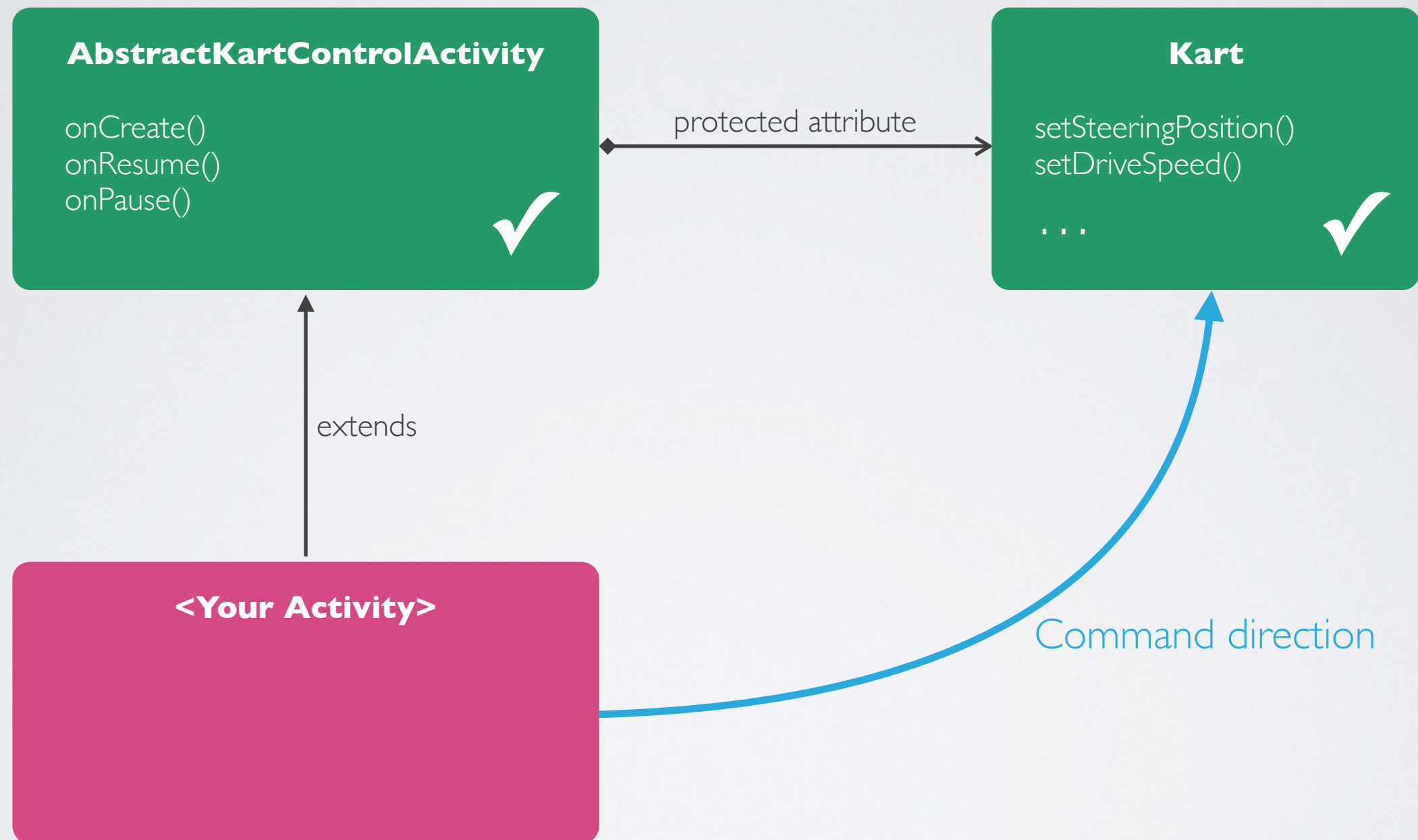


Remote Control Android App



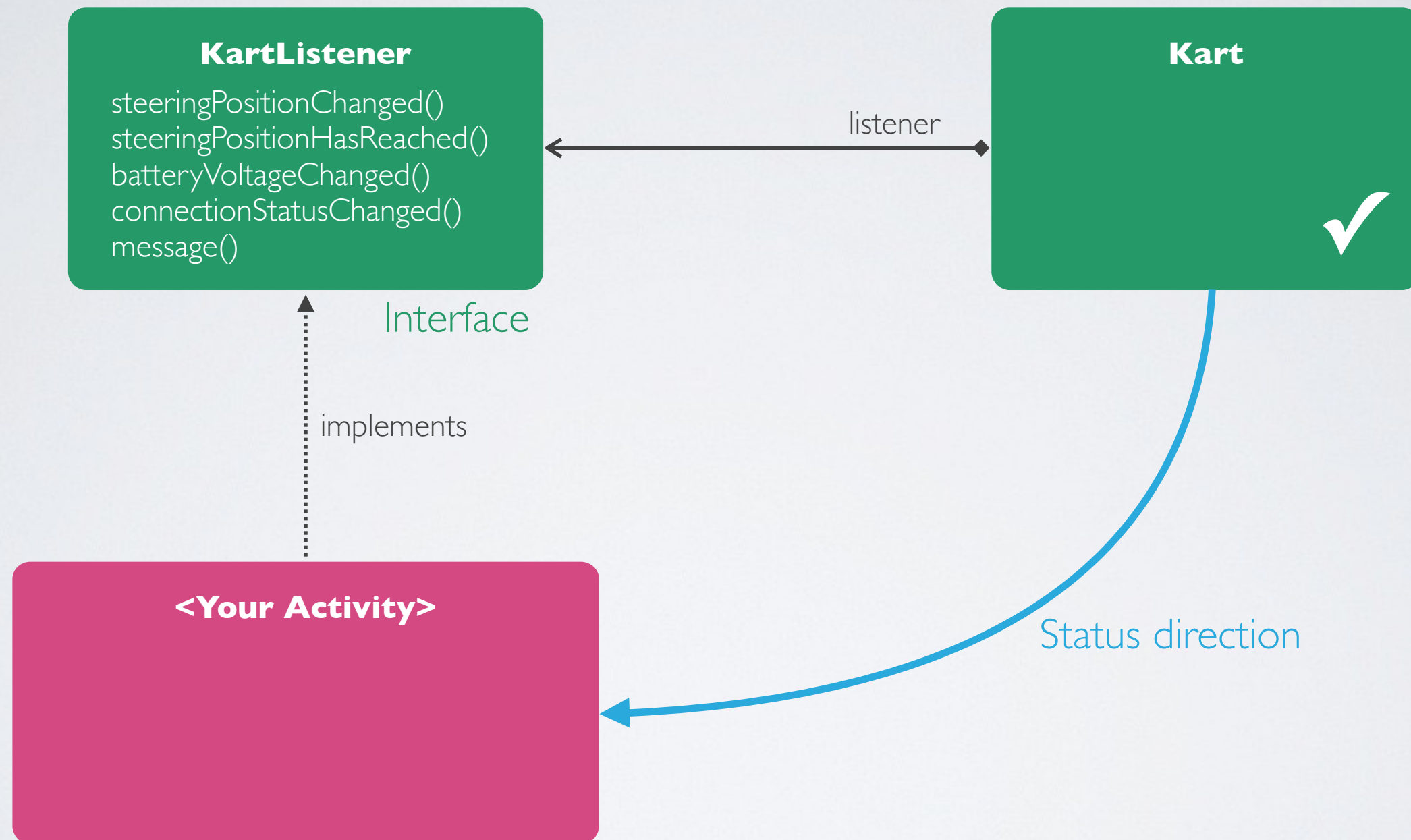


Remote Control Android App



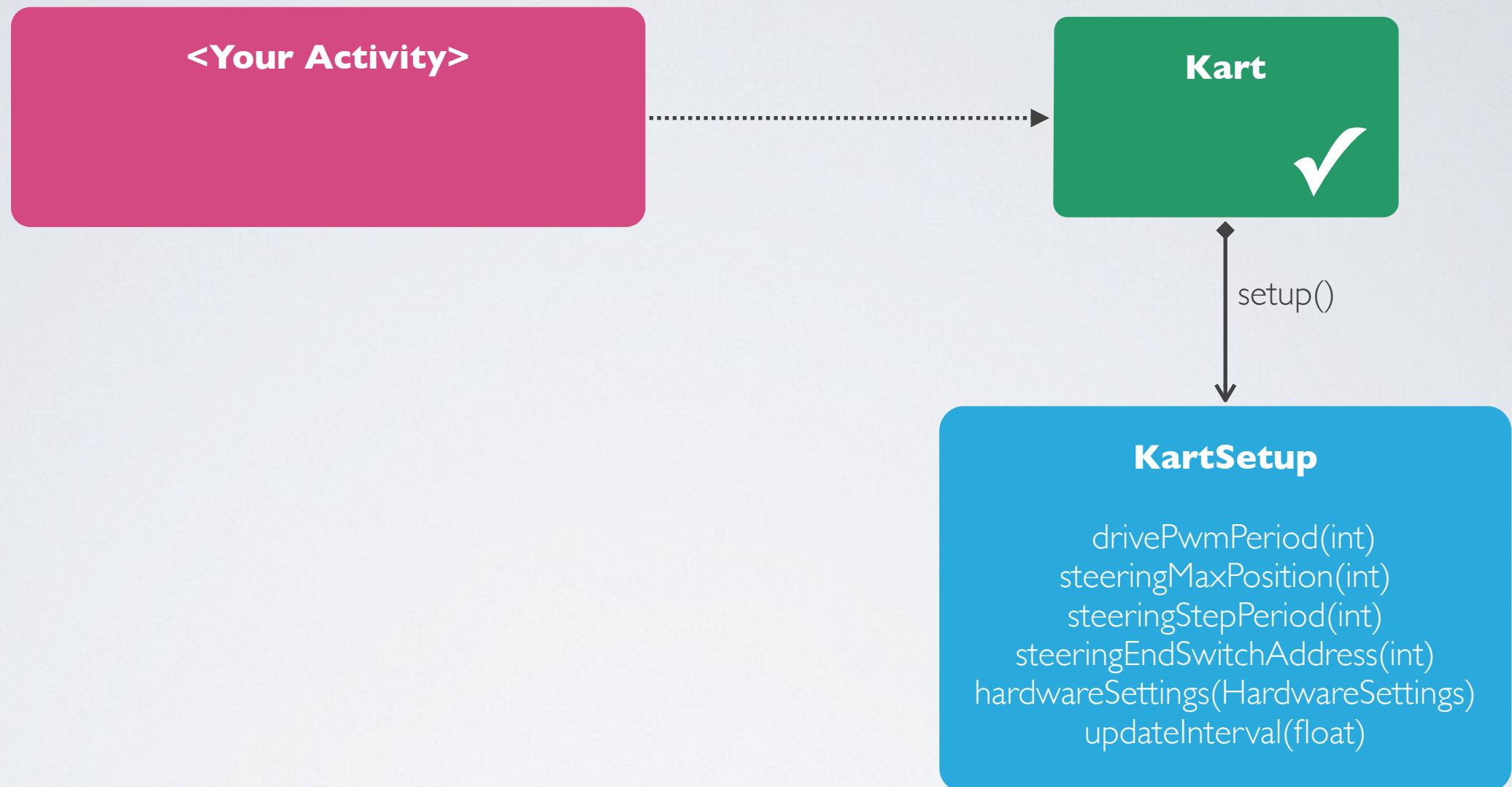


Remote Control Android App





Remote Control Android App



```
void onCreate(Bundle savedInstanceState) {  
    ...  
    kart.setup().drivePwmPeriod(50).steeringStepPeriod(25).updateInterval(100);  
    ...  
}
```




- **Slider control**
 - Direction
 - Speed
- **Progress Bar status**
 - Battery level
 - Steering position
- **Accelerometer (Orientation) control**
 - Button to enable orientation control
 - Device orientation controls sliders or kart



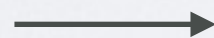
Software Grade



Functional blackbox tests
during morning of the last
day

All mandatory features

Direction control
Speed control
Battery display
Direction display



4.0

+

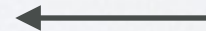
0.5

=

Grade

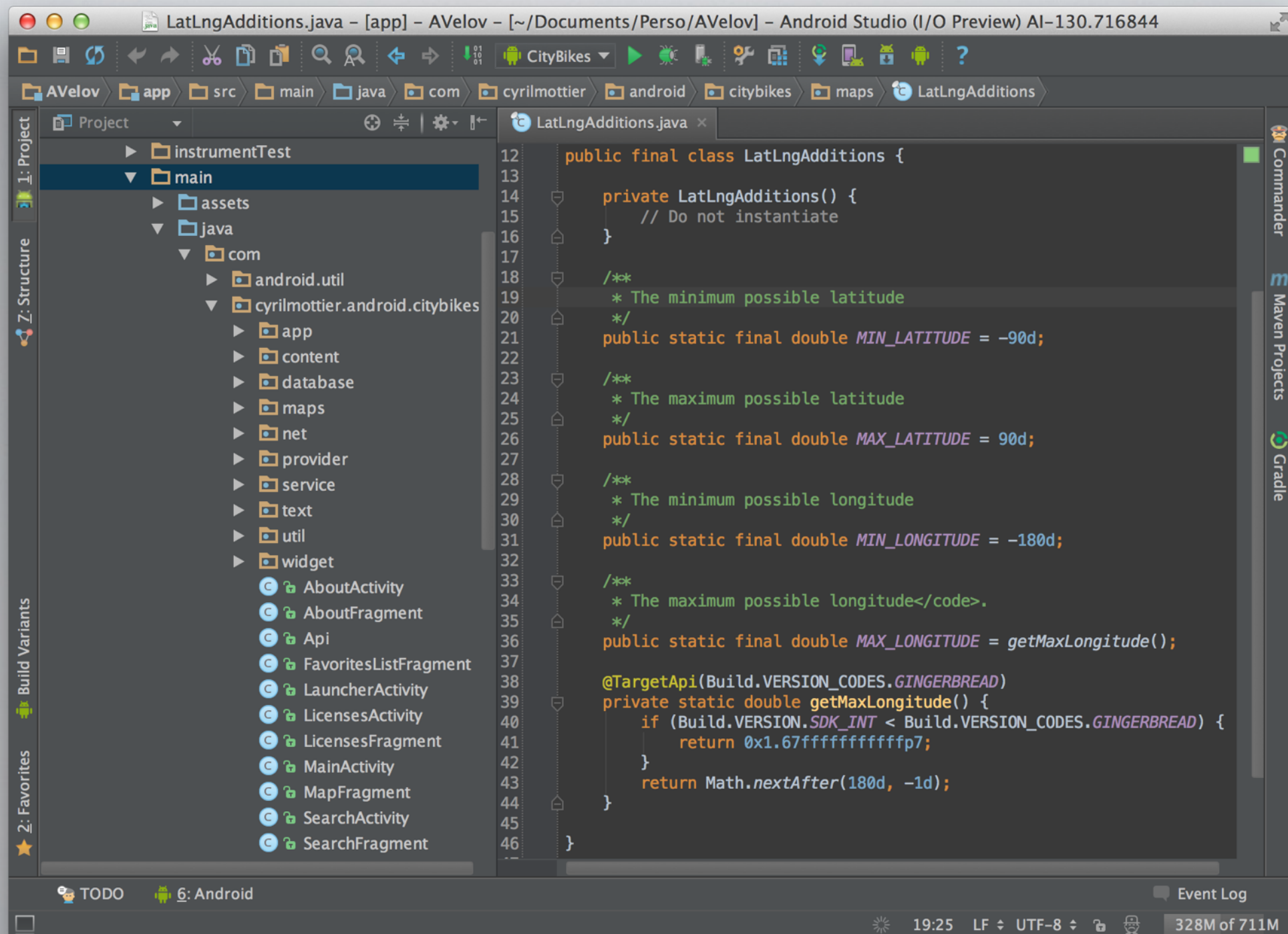
Per optional feature

LED's
Ultrasound sensor
Proximity sensor
Hall sensor
Steering assistant
Other improvements





Android Studio



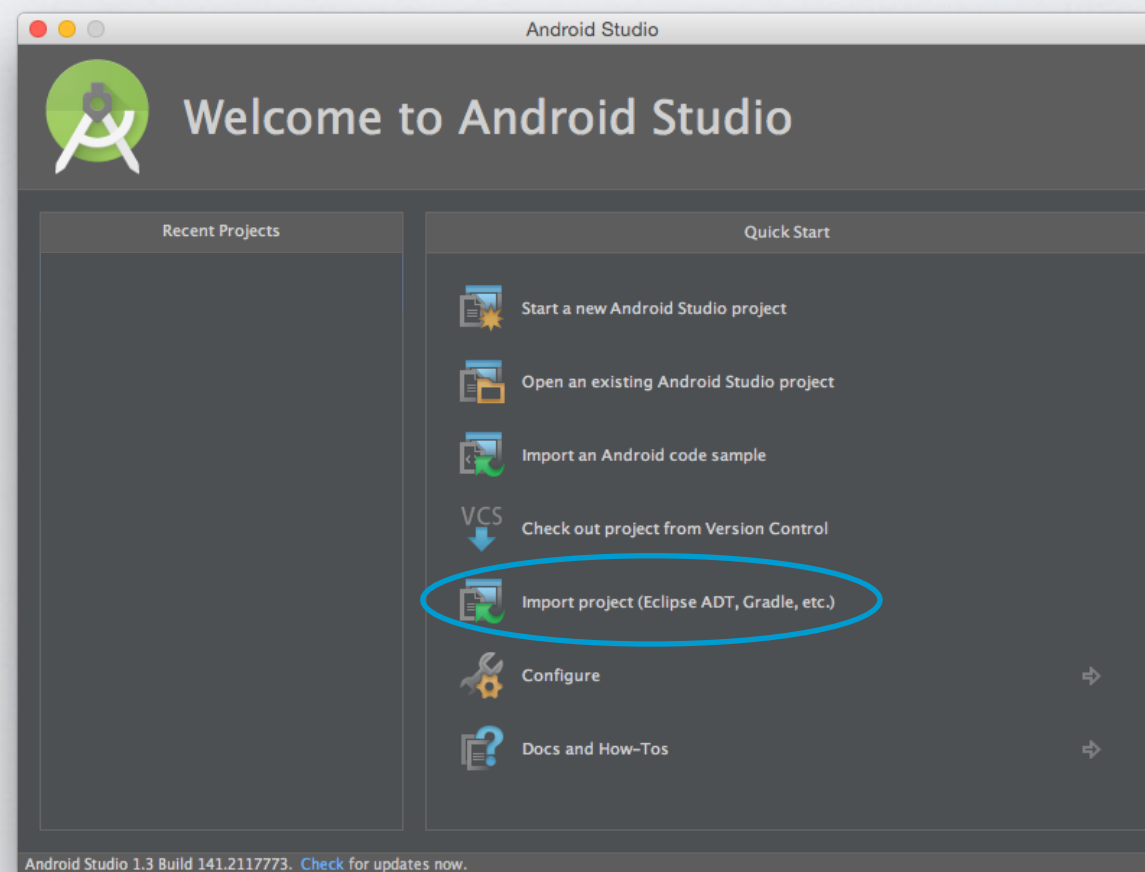


Android Studio

Download the Kart project template from the wiki

Extract the archive to the local disk

Import the extracted folder in Android Studio



Galaxy Nexus

- Connect Phone to PC's USB port
- Power on the phone
- Use default configurations during setup wizard
- Enable developer mode:
Go to Settings > About Phone and press 7 times on „Build Number“
- Enable USB debugging:
Go to Settings > Developer options and check „USB debugging“
- Install and start your Android application:
 - In Android Studio, press green play button and select Galaxy Nexus phone in the list.
 - On the phone, answer yes to allow USB debugging in popup.
 - Now your application should be running on the phone.



Download **Kart.apk** from
<http://wiki.hevs.ch/fsi/index.php5/Kart>

```
C:\Users\uadmin>cd "c:\Program Files (x86)\Android\android-sdk\platform-tools\"  
c:\Program Files (x86)\Android\android-sdk\platform-tools>adb install c:\Users\uadmin\Downloads\Kart.apk  
3979 KB/s (937315 bytes in 0.230s)  
  pkg: /data/local/tmp/Kart.apk  
Success  
c:\Program Files (x86)\Android\android-sdk\platform-tools>
```