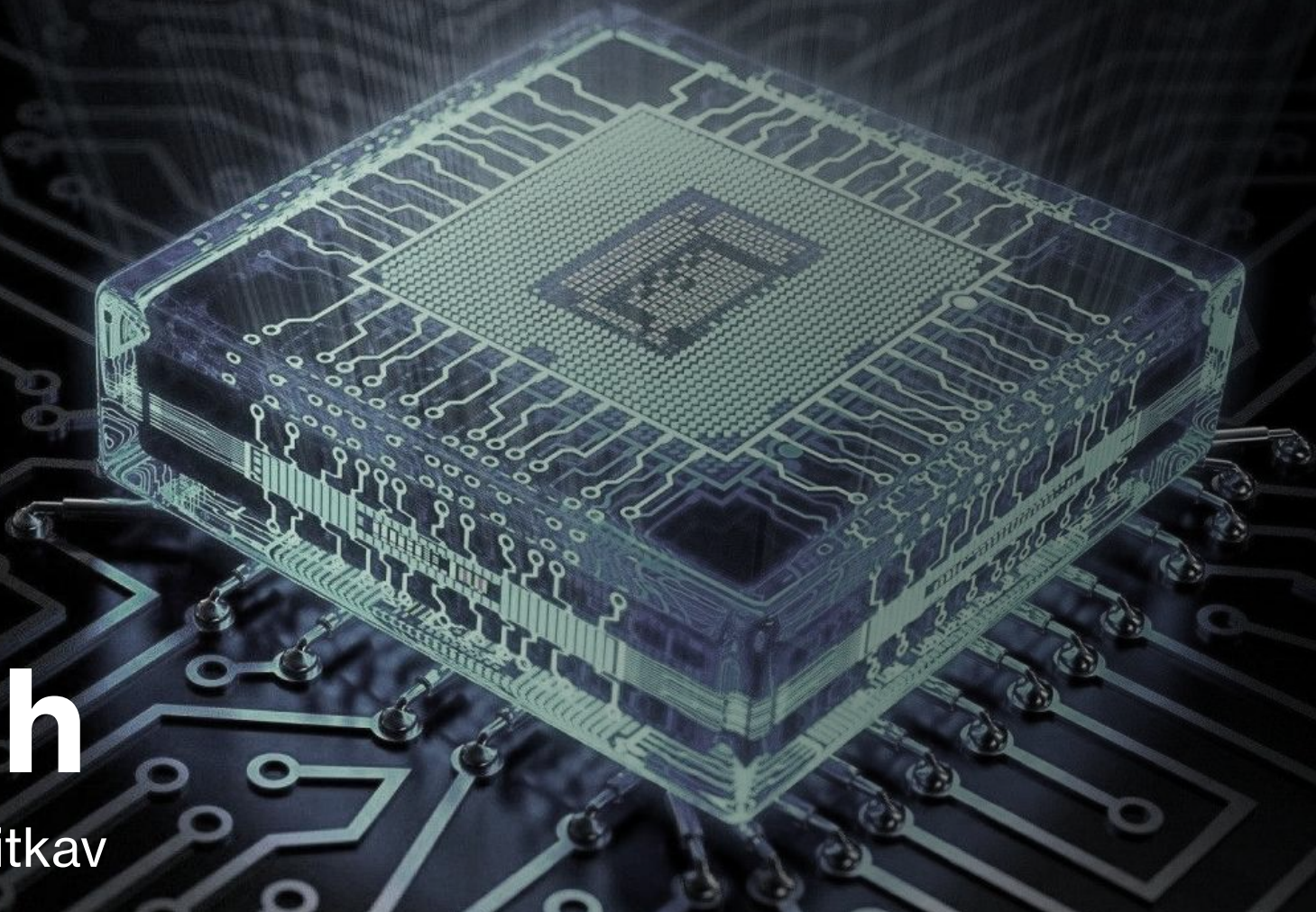


RoboWatch

Andrej Repaský, Sebastian Ritkav



```
144  
145 if (State == 1) Rotate(91, true, true);  
146 if (State == 2) Rotate(91, false, true);  
147 if (State == 3) Rotate(180, false, true);  
148 } else {  
149 MoveMotors(true, false, false, 0);  
150 }  
151  
152 if (State == 4) MoveMotors(true, false, false, 0);  
153
```

```
154  
155 ///color files  
156 if (ColorD != PrevColorD) {  
157 PrevColorD = ColorD;  
158  
159 ///blue file  
160 if (ColorD == 2 && CanBlue) {  
161 DoBlue(90);  
162 State = 4;  
163 MoveMotors(true, false, false, 0);  
164 delay(5500);  
165 }  
166  
167 if (ColorD != 2) {  
168 CanBlue = true;  
169 }  
170  
171 ///Black tile  
172 if (ColorD == 1) {  
173 DoBlackC();  
174 }  
175 }  
176  
177  
178 ///pohyb  
179 if (!CanMoveBlack) {  
180 if (PrevDistF >= DistF + 280) {  
181 CanMove = false;  
182 MoveMotors(true, false, false, 0);  
183
```

Návrh robota

Skúška senzorov

Programovanie

1.

3.

5.



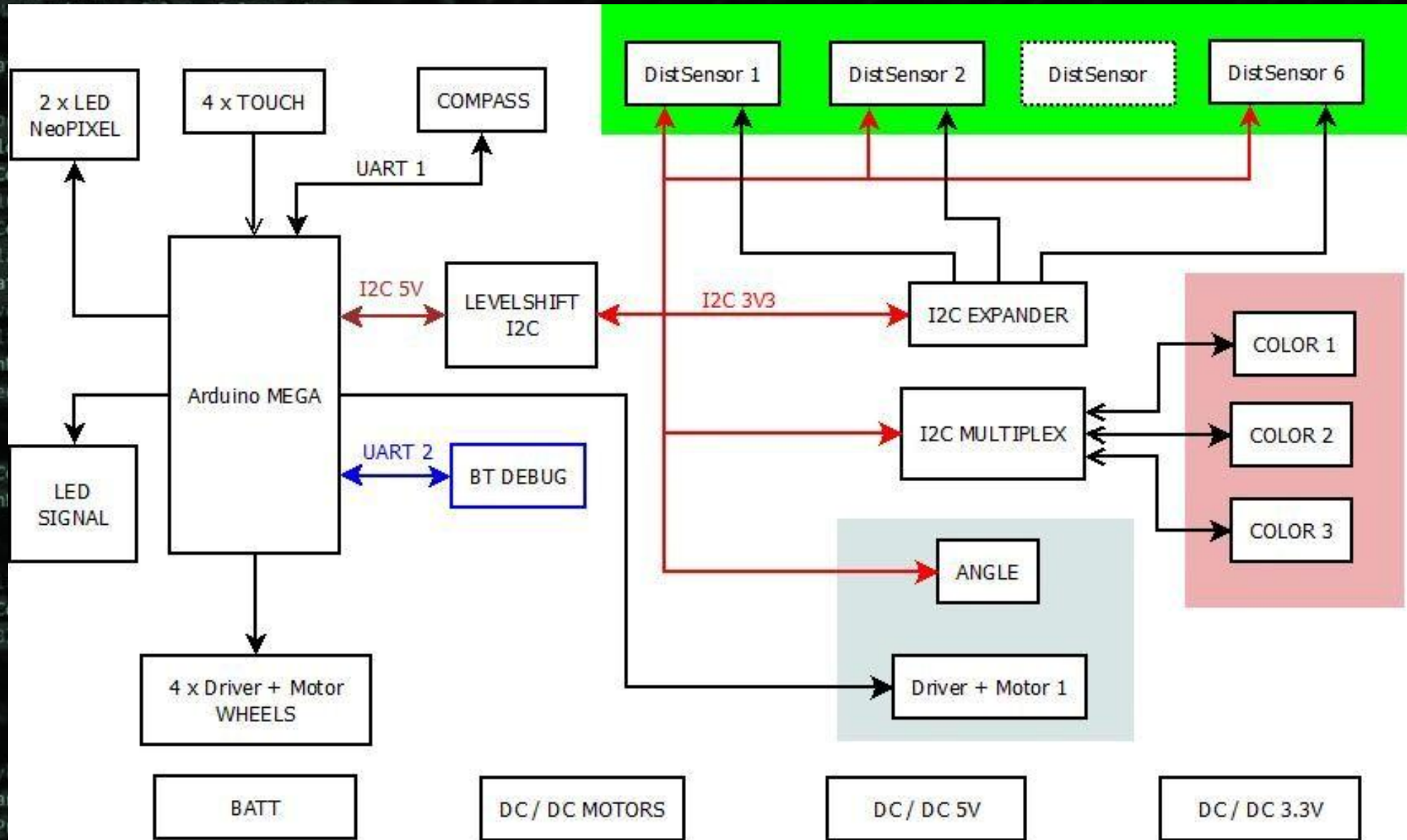
2.

4.

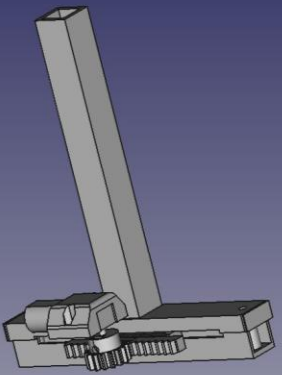
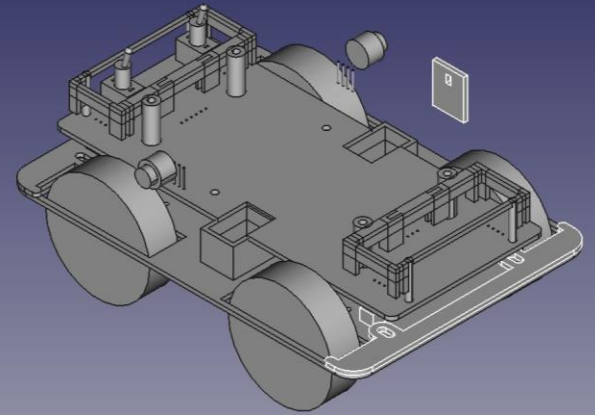
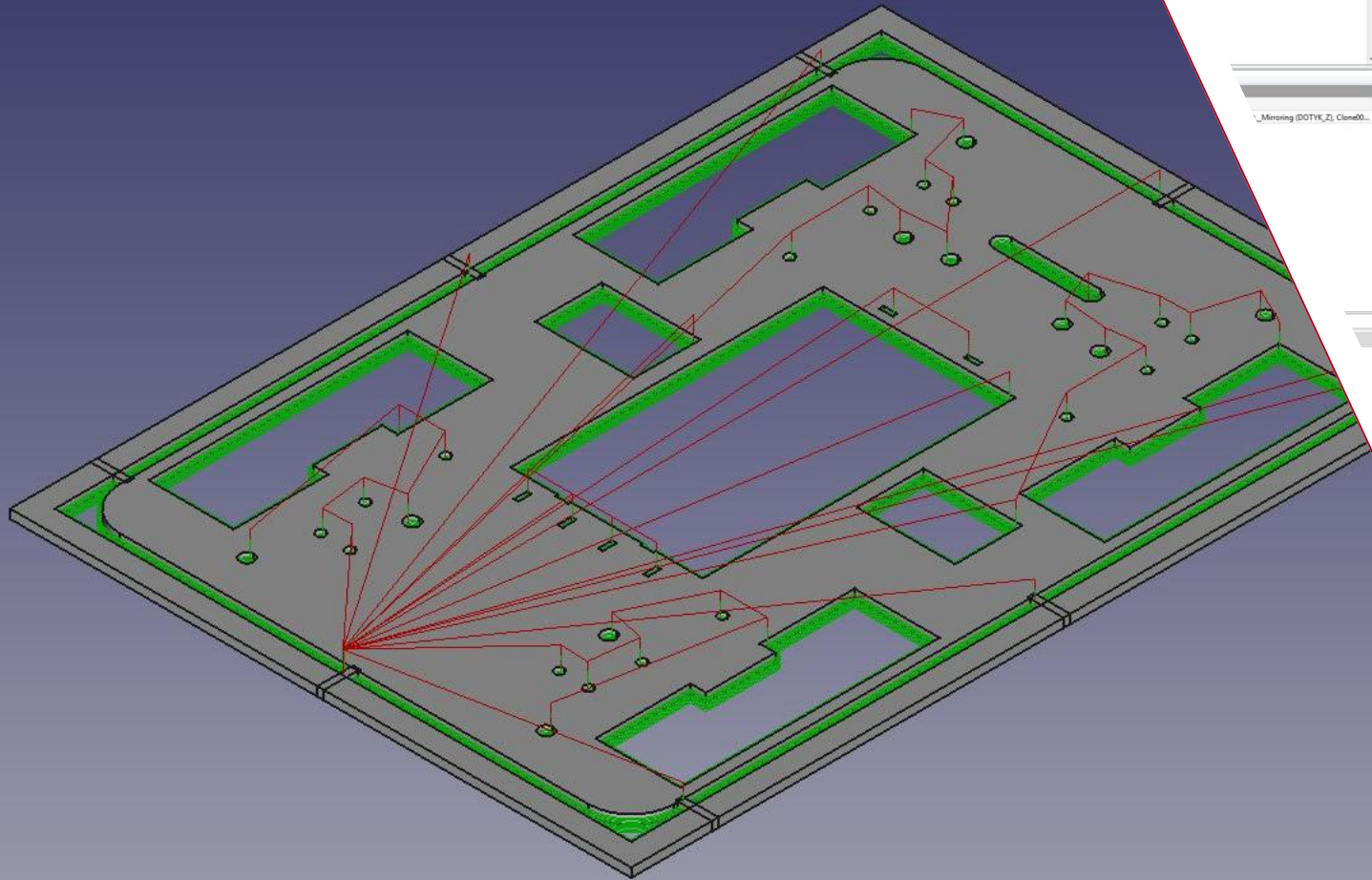
Vytlačenie robota

Zapojenie robota

Schéma



Návrh



[Makro]

Úvodná stránka ROBOT_MAZE_2025_v10 : 1*

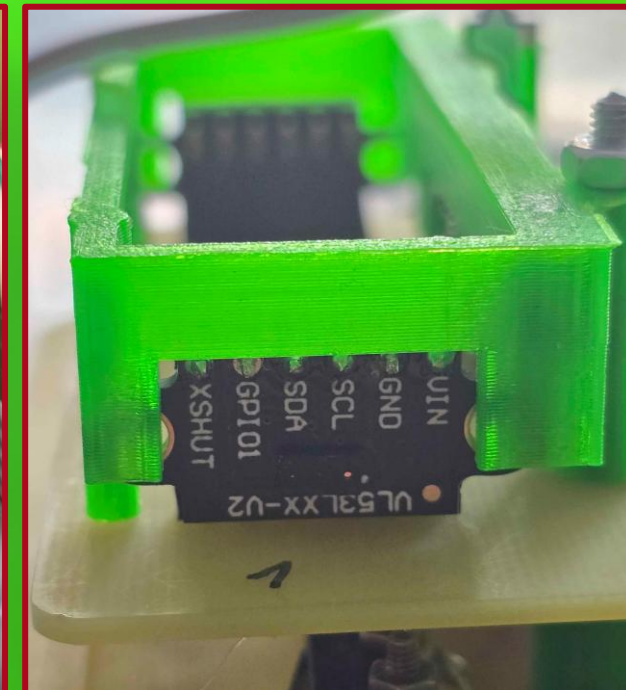
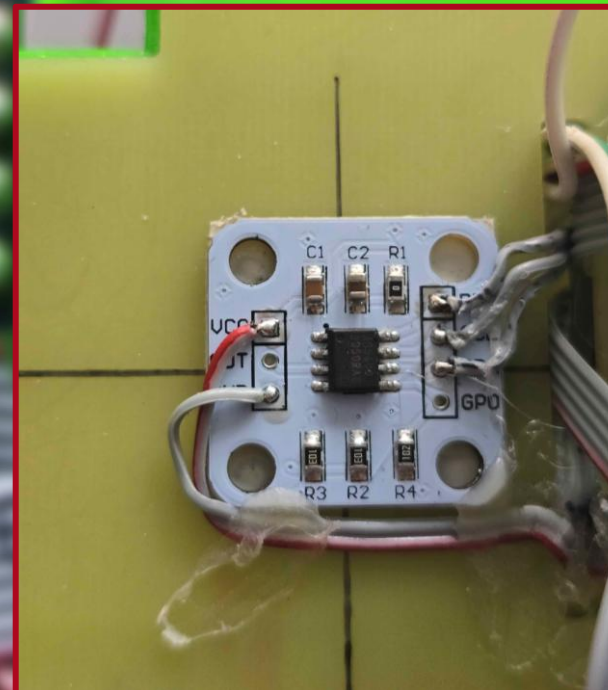
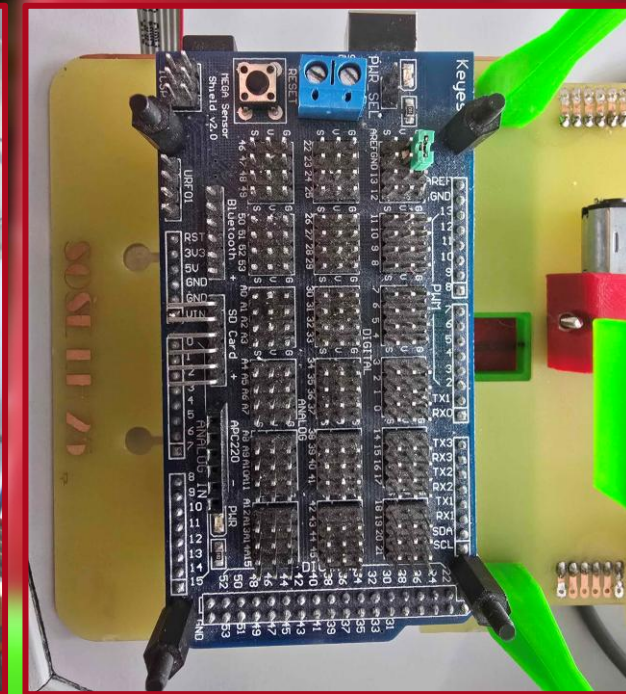
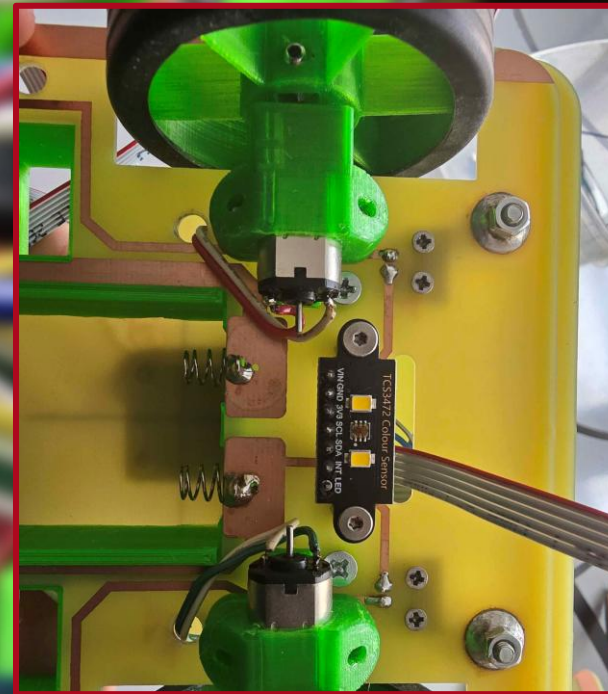
CAD 290,99 mm x

EN 10

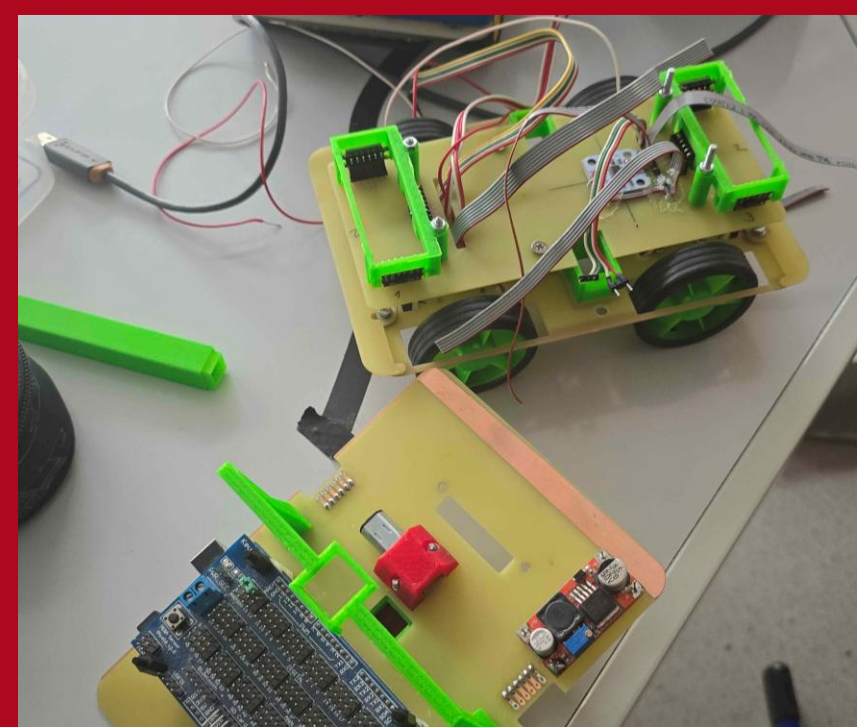
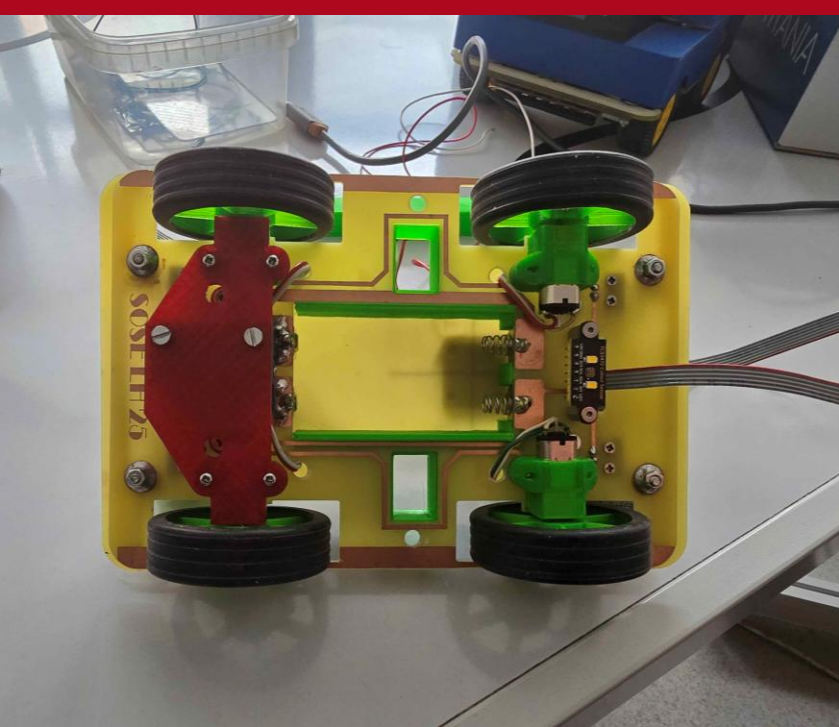
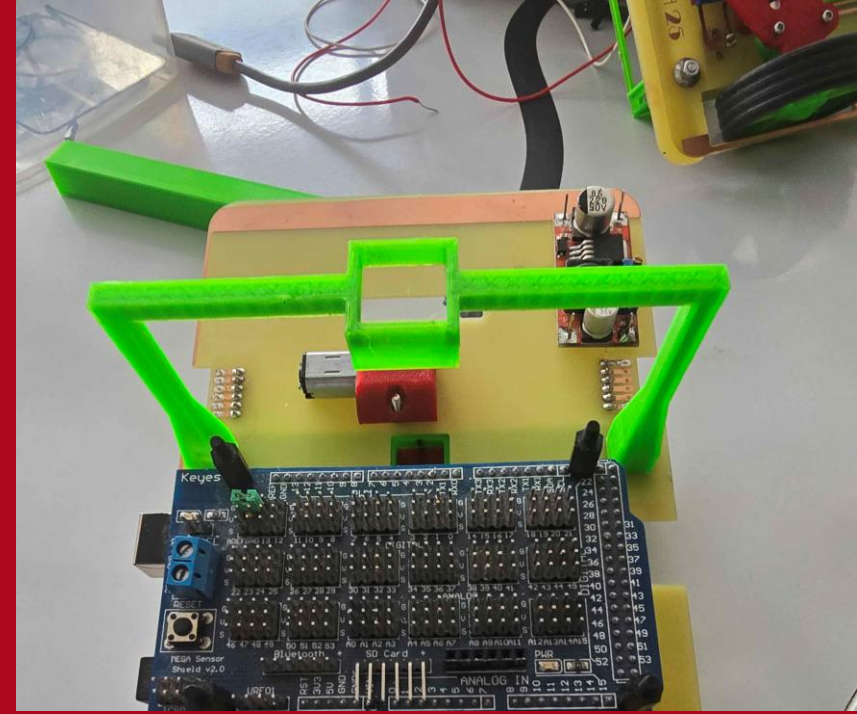
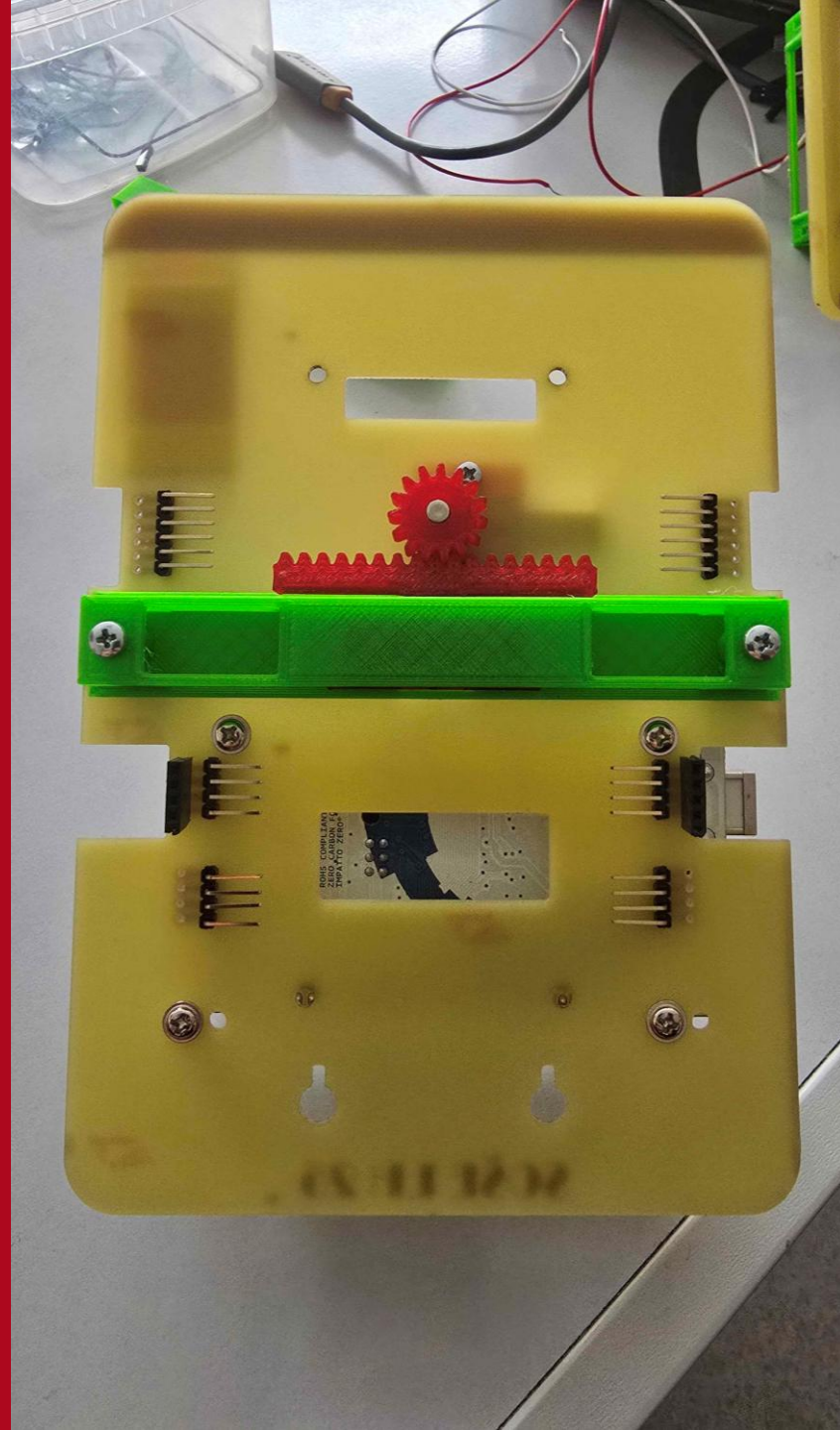
Hardware

Použili sme:

- Arduino Mega
- Sensory
 - 6x Laser senzor VL53LOX-V2 → použité na meranie vzdialenosti
 - 4x Color senzor TCS34725 → použité na rozpoznávanie farby obetí
 - 1x Magnetický senzor AS5600 → použité na meranie rotácie vyhadzovacieho mechanizmu pre záchranné balíčky
 - 1x Gyro senzor GY-25 → použité na meranie rotácie robota
 - 4x Tlačítko → použité na snímanie nárazu
- 5x Mikro motor s prevodovkou N20 → použité na pohyb robota
- 1x Stabilizátor XL6009
- 2x Mini DC stabilizátor
- 1x I2C Multiplexer TCA9548A
- 1x I2C Expander PCF8574T
- Externé nasvietenie Color senzorov



Zostavenie



Software

- Programovali sme v Arduino IDE
- **Použili sme tieto knižnice:**
 - Wire.h
 - Adafruit_TCS34725.h
 - Arduino.h
 - TCA9548A.h
 - PCF8574.h
 - Adafruit_VL53L0X.h
 - Adafruit_NeoPixel.h
 - avr/power.h
- **Riešili sme:**
 - Správny pohyb robota
 - Správne čítanie senzorov
 - Snímanie obetí
 - Vyhadzovanie záchranných balíčkov

Otáčanie robota:

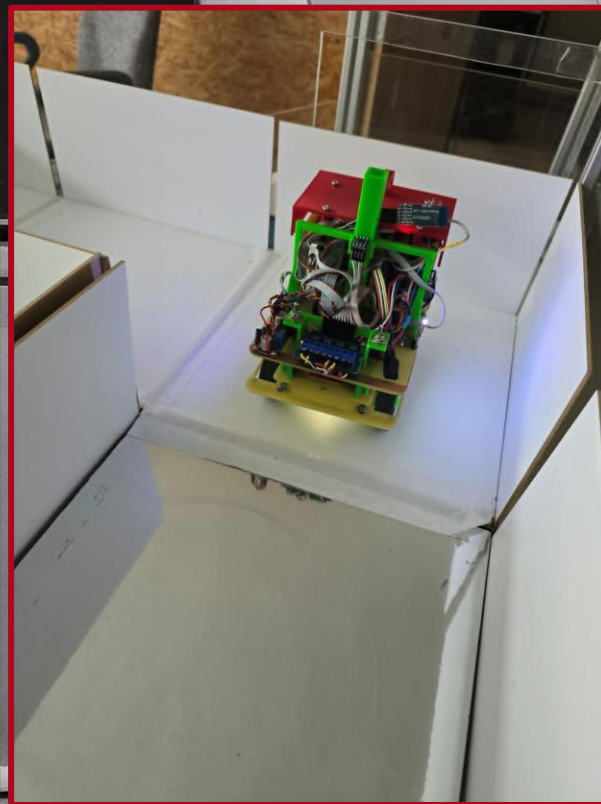
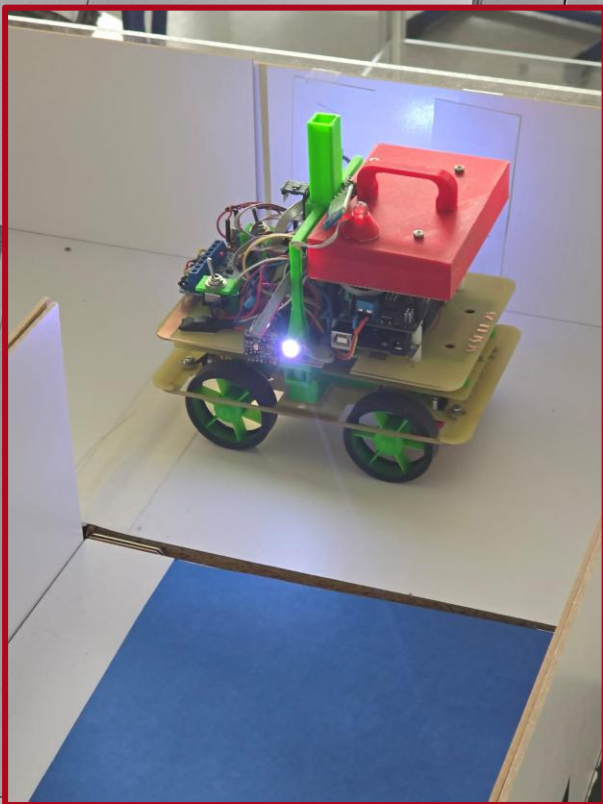
```
506 void Rotate(float Angle, bool Left, bool Settle) {
507     float delta = Yaw - LYaw;
508     if (delta > 180) YawPlus -= 1;
509     if (delta < -180) YawPlus += 1;
510     CYaw = Yaw + (360 * YawPlus);
511     LYaw = Yaw;
512
513     if (!Left) {
514         if (!IsTurningRight) {
515             SYaw = CYaw;
516             IsTurningRight = true;
517         }
518         if (CYaw > SYaw + Angle) {
519             State = 0;
520             IsTurningRight = false;
521             if (Settle) {
522                 IsSettling = true;
523                 SettlingLeft = false;
524             }
525         } else {
526             MoveMotors(true, false, true, 80);
527         }
528     } else {
529         if (!IsTurningLeft) {
530             SYaw = CYaw;
531             IsTurningLeft = true;
532         }
533         // Read();
534         if (CYaw < SYaw - Angle) {
535             State = 0;
536             IsTurningLeft = false;
537             if (Settle) {
538                 IsSettling = true;
539                 SettlingLeft = true;
540             }
541         } else {
542             MoveMotors(true, true, false, 80);
543         }
544     }
545 }
546 }
547 }
```

Dorovňovanie robota:

```
713 void RunSettling() {
714     if (!SettlingLeft) {
715         PPL.rangingTest(&PP, false);
716         DistFR = PP.RangeMilliMeter;
717
718         ZPL.rangingTest(&PZ, false);
719         DistRR = PZ.RangeMilliMeter;
720
721         if (DistFR > DistRR + 15) {
722             MoveMotors(true, !false, !true, 60);
723         } else {
724             // State = 0;
725             // CanStopFront = true;
726             // x = false;
727             ReverseSettle();
728         }
729     } else {
730         PLL.rangingTest(&pL, false);
731         DistFL = pL.RangeMilliMeter;
732
733         ZLL.rangingTest(&LZ, false);
734         DistRL = LZ.RangeMilliMeter;
735         if (DistFL > DistRL + 20) {
736             MoveMotors(true, !true, !false, 60);
737         } else {
738             // State = 0;
739             // CanStopFront = true;
740             // x = false;
741             ReverseSettle();
742         }
743     }
744     // delay(10);
745 }
746 }
```

Testovanie

- Vybudovali sme cvičné bludisko
- Nakreslili sme si znaky, ktoré predstavujú obete
- Simulovali sme jazdu v prostredí a dávali sme robotovi rôzne prekážky



RoboWatch

```
444 if (State == 1) Rotate(91,  
445 if (State == 2) Rotate(91,  
446 if (State == 3) Rotate(180,  
447 } else {  
448 MoveMotors(true, false, fal  
449 }  
450 if (State == 4) MoveMotors(tr  
451  
452 //color  
453 if (Color == PrevColorD) {  
454 PrevCo  
455 ColorD;  
456 }  
457 //blue  
458 if (Color == 2 && CanBlue)  
459 del  
460 ;  
461 State  
462 MoveMotors(true, false, f  
463 del  
464 );  
465 Can  
466 false;  
467 Can  
468 (false);  
469 }  
470 }  
471 if (Color == 2) {  
472 CanB  
473 true;  
474 }  
475 }  
476 }  
477 }  
478 }  
479 }  
480 }  
481 }  
482 }  
483 }  
484 }  
485 }  
486 }  
487 }  
488 }  
489 }  
490 }  
491 }  
492 }  
493 }  
494 }  
495 }  
496 }  
497 }  
498 }  
499 }  
500 }  
501 }  
502 }  
503 }  
504 }  
505 }  
506 }  
507 }  
508 }  
509 }  
510 }  
511 }  
512 }  
513 }  
514 }  
515 }  
516 }  
517 }  
518 }  
519 }  
520 }  
521 }  
522 }  
523 }  
524 }  
525 }  
526 }  
527 }  
528 }  
529 }  
530 }  
531 }  
532 }  
533 }  
534 }  
535 }  
536 }  
537 }  
538 }  
539 }  
540 }  
541 }  
542 }  
543 }  
544 }  
545 }  
546 }  
547 }  
548 }  
549 }  
550 }  
551 }  
552 }  
553 }  
554 }  
555 }  
556 }  
557 }  
558 }  
559 }  
560 }  
561 }  
562 }  
563 }  
564 }  
565 }  
566 }  
567 }  
568 }  
569 }  
570 }  
571 }  
572 }  
573 }  
574 }  
575 }  
576 }  
577 }  
578 }  
579 }  
580 }  
581 }  
582 }  
583 }  
584 }  
585 }  
586 }  
587 }  
588 }  
589 }  
590 }  
591 }  
592 }  
593 }  
594 }  
595 }  
596 }  
597 }  
598 }  
599 }  
600 }  
601 }  
602 }  
603 }  
604 }  
605 }  
606 }  
607 }  
608 }  
609 }  
610 }  
611 }  
612 }  
613 }  
614 }  
615 }  
616 }  
617 }  
618 }  
619 }  
620 }  
621 }  
622 }  
623 }  
624 }  
625 }  
626 }  
627 }  
628 }  
629 }  
630 }  
631 }  
632 }  
633 }  
634 }  
635 }  
636 }  
637 }  
638 }  
639 }  
640 }  
641 }  
642 }  
643 }  
644 }  
645 }  
646 }  
647 }  
648 }  
649 }  
650 }  
651 }  
652 }  
653 }  
654 }  
655 }  
656 }  
657 }  
658 }  
659 }  
660 }  
661 }  
662 }  
663 }  
664 }  
665 }  
666 }  
667 }  
668 }  
669 }  
670 }  
671 }  
672 }  
673 }  
674 }  
675 }  
676 }  
677 }  
678 }  
679 }  
680 }  
681 }  
682 }  
683 }  
684 }  
685 }  
686 }  
687 }  
688 }  
689 }  
690 }  
691 }  
692 }  
693 }  
694 }  
695 }  
696 }  
697 }  
698 }  
699 }  
700 }  
701 }  
702 }  
703 }  
704 }  
705 }  
706 }  
707 }  
708 }  
709 }  
710 }  
711 }  
712 }  
713 }  
714 }  
715 }  
716 }  
717 }  
718 }  
719 }  
720 }  
721 }  
722 }  
723 }  
724 }  
725 }  
726 }  
727 }  
728 }  
729 }  
730 }  
731 }  
732 }  
733 }  
734 }  
735 }  
736 }  
737 }  
738 }  
739 }  
740 }  
741 }  
742 }  
743 }  
744 }  
745 }  
746 }  
747 }  
748 }  
749 }  
750 }  
751 }  
752 }  
753 }  
754 }  
755 }  
756 }  
757 }  
758 }  
759 }  
760 }  
761 }  
762 }  
763 }  
764 }  
765 }  
766 }  
767 }  
768 }  
769 }  
770 }  
771 }  
772 }  
773 }  
774 }  
775 }  
776 }  
777 }  
778 }  
779 }  
780 }  
781 }  
782 }  
783 }  
784 }  
785 }  
786 }  
787 }  
788 }  
789 }  
790 }  
791 }  
792 }  
793 }  
794 }  
795 }  
796 }  
797 }  
798 }  
799 }  
800 }  
801 }  
802 }  
803 }  
804 }  
805 }  
806 }  
807 }  
808 }  
809 }  
810 }  
811 }  
812 }  
813 }  
814 }  
815 }  
816 }  
817 }  
818 }  
819 }  
820 }  
821 }  
822 }  
823 }  
824 }  
825 }  
826 }  
827 }  
828 }  
829 }  
830 }  
831 }  
832 }  
833 }  
834 }  
835 }  
836 }  
837 }  
838 }  
839 }  
840 }  
841 }  
842 }  
843 }  
844 }  
845 }  
846 }  
847 }  
848 }  
849 }  
850 }  
851 }  
852 }  
853 }  
854 }  
855 }  
856 }  
857 }  
858 }  
859 }  
860 }  
861 }  
862 }  
863 }  
864 }  
865 }  
866 }  
867 }  
868 }  
869 }  
870 }  
871 }  
872 }  
873 }  
874 }  
875 }  
876 }  
877 }  
878 }  
879 }  
880 }  
881 }  
882 }  
883 }  
884 }  
885 }  
886 }  
887 }  
888 }  
889 }  
890 }  
891 }  
892 }  
893 }  
894 }  
895 }  
896 }  
897 }  
898 }  
899 }  
900 }  
901 }  
902 }  
903 }  
904 }  
905 }  
906 }  
907 }  
908 }  
909 }  
910 }  
911 }  
912 }  
913 }  
914 }  
915 }  
916 }  
917 }  
918 }  
919 }  
920 }  
921 }  
922 }  
923 }  
924 }  
925 }  
926 }  
927 }  
928 }  
929 }  
930 }  
931 }  
932 }  
933 }  
934 }  
935 }  
936 }  
937 }  
938 }  
939 }  
940 }  
941 }  
942 }  
943 }  
944 }  
945 }  
946 }  
947 }  
948 }  
949 }  
950 }  
951 }  
952 }  
953 }  
954 }  
955 }  
956 }  
957 }  
958 }  
959 }  
960 }  
961 }  
962 }  
963 }  
964 }  
965 }  
966 }  
967 }  
968 }  
969 }  
970 }  
971 }  
972 }  
973 }  
974 }  
975 }  
976 }  
977 }  
978 }  
979 }  
980 }  
981 }  
982 }  
983 }  
984 }  
985 }  
986 }  
987 }  
988 }  
989 }  
990 }  
991 }  
992 }  
993 }  
994 }  
995 }  
996 }  
997 }  
998 }  
999 }  
1000 }
```

