```c
/* USER CODE BEGIN Header */
  * @file           : main.c
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "APA102C.h"
#include "MAX6955.h"

/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */

/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/
I2C_HandleTypeDef hi2c2;

/* USER CODE BEGIN PV */
uint8_t buf[15];
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C2_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
void SysTickDelayCount2(unsigned long ulCount){
    CoreDebug->DEMCR |= CoreDebug_DEMCR_TRCENA_Msk;
    ITM->LAR = 0xC5ACCE55;
    DWT->CYCCNT = 0;
    DWT->CTRL |= DWT_CTRL_CYCCNTENA_Msk;

    while(DWT->CYCCNT < ulCount);
}
/*
void Meten(){
    HAL_GPIO_WritePin(SPI1_CS_GPIO_Port,SPI1_CS_Pin,GPIO_PIN_RESET);
    SysTickDelayCount2(t);
```

```
 74    HAL_GPIO_WritePin(SPI1_CS_GPIO_Port,SPI1_CS_Pin,GPIO_PIN_SET);
 75    for (int8_t j = 0; j < 1; j++){
 76        for (int8_t i = 0; i < 8; i++){
 77            SysTickDelayCount2((t/2));
 78            buf[j][i] = (HAL_GPIO_ReadPin(SPI1_MISO_GPIO_Port,SPI1_MISO_Pin) ^ 1); //Inverteer
   de lezing
 79            SysTickDelayCount2((t/2));
 80            HAL_GPIO_WritePin(SPI1_CLK_GPIO_Port,SPI1_CLK_Pin,GPIO_PIN_SET);
 81            SysTickDelayCount2(t);
 82            HAL_GPIO_WritePin(SPI1_CLK_GPIO_Port,SPI1_CLK_Pin,GPIO_PIN_RESET);
 83        }
 84    }
 85}
 86*/
 87/* USER CODE END 0 */
 88
 89/**
 90  * @brief  The application entry point.
 91  * @retval int
 92  */
 93int main(void)
 94{
 95  /* USER CODE BEGIN 1 */
 96
 97  /* USER CODE END 1 */
 98
 99  /* MCU Configuration--------------------------------------------------------*/
100
101  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
102  HAL_Init();
103
104  /* USER CODE BEGIN Init */
105
106  /* USER CODE END Init */
107
108  /* Configure the system clock */
109  SystemClock_Config();
110
111  /* USER CODE BEGIN SysInit */
112
113  /* USER CODE END SysInit */
114
115  /* Initialize all configured peripherals */
116  MX_GPIO_Init();
117  MX_I2C2_Init();
118  /* USER CODE BEGIN 2 */
119
120  // 7-segment aansturen in test stand
121      SS_Start(1);
122
123  // APA102C's aansturen
124      LED_Start();
125      LED_Rood(10);
126      LED_Groen(10);
127      LED_Blauw(10);
128      LED_RGB(0, 10, 10);
129      LED_Start();
```

```
130
131   /* USER CODE END 2 */
132
133   /* Infinite loop */
134   /* USER CODE BEGIN WHILE */
135   while (1)
136   {
137     /* USER CODE END WHILE */
138
139     /* USER CODE BEGIN 3 */
140   }
141   /* USER CODE END 3 */
142 }
143
144 /**
145   * @brief System Clock Configuration
146   * @retval None
147   */
148 void SystemClock_Config(void)
149 {
150   RCC_OscInitTypeDef RCC_OscInitStruct = {0};
151   RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
152
153   /** Initializes the RCC Oscillators according to the specified parameters
154   * in the RCC_OscInitTypeDef structure.
155   */
156   RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
157   RCC_OscInitStruct.HSIState = RCC_HSI_ON;
158   RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
159   RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
160   RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI_DIV2;
161   RCC_OscInitStruct.PLL.PLLMUL = RCC_PLL_MUL9;
162   if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
163   {
164     Error_Handler();
165   }
166   /** Initializes the CPU, AHB and APB buses clocks
167   */
168   RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
169                                 |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
170   RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
171   RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
172   RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
173   RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;
174
175   if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_1) != HAL_OK)
176   {
177     Error_Handler();
178   }
179 }
180
181 /**
182   * @brief I2C2 Initialization Function
183   * @param None
184   * @retval None
185   */
186 static void MX_I2C2_Init(void)
```

```
187 {
188
189   /* USER CODE BEGIN I2C2_Init 0 */
190
191   /* USER CODE END I2C2_Init 0 */
192
193   /* USER CODE BEGIN I2C2_Init 1 */
194
195   /* USER CODE END I2C2_Init 1 */
196   hi2c2.Instance = I2C2;
197   hi2c2.Init.ClockSpeed = 100000;
198   hi2c2.Init.DutyCycle = I2C_DUTYCYCLE_2;
199   hi2c2.Init.OwnAddress1 = 0;
200   hi2c2.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
201   hi2c2.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
202   hi2c2.Init.OwnAddress2 = 0;
203   hi2c2.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
204   hi2c2.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
205   if (HAL_I2C_Init(&hi2c2) != HAL_OK)
206   {
207     Error_Handler();
208   }
209   /* USER CODE BEGIN I2C2_Init 2 */
210
211   /* USER CODE END I2C2_Init 2 */
212
213 }
214
215 /**
216   * @brief GPIO Initialization Function
217   * @param None
218   * @retval None
219   */
220 static void MX_GPIO_Init(void)
221 {
222   GPIO_InitTypeDef GPIO_InitStruct = {0};
223
224   /* GPIO Ports Clock Enable */
225   __HAL_RCC_GPIOC_CLK_ENABLE();
226   __HAL_RCC_GPIOA_CLK_ENABLE();
227   __HAL_RCC_GPIOB_CLK_ENABLE();
228
229   /*Configure GPIO pin Output Level */
230   HAL_GPIO_WritePin(SPI1_CS_GPIO_Port, SPI1_CS_Pin, GPIO_PIN_RESET);
231
232   /*Configure GPIO pin Output Level */
233   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2|SPI1_CLK_Pin, GPIO_PIN_RESET);
234
235   /*Configure GPIO pin Output Level */
236   HAL_GPIO_WritePin(GPIOB, SPI2_SCK_Pin|SPI2_MOSI_Pin, GPIO_PIN_RESET);
237
238   /*Configure GPIO pin : SPI1_CS_Pin */
239   GPIO_InitStruct.Pin = SPI1_CS_Pin;
240   GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
241   GPIO_InitStruct.Pull = GPIO_NOPULL;
242   GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
243   HAL_GPIO_Init(SPI1_CS_GPIO_Port, &GPIO_InitStruct);
```

```c

  /*Configure GPIO pins : PA2 SPI1_CLK_Pin */
  GPIO_InitStruct.Pin = GPIO_PIN_2|SPI1_CLK_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

  /*Configure GPIO pin : SPI1_MISO_Pin */
  GPIO_InitStruct.Pin = SPI1_MISO_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(SPI1_MISO_GPIO_Port, &GPIO_InitStruct);

  /*Configure GPIO pins : SPI2_SCK_Pin SPI2_MOSI_Pin */
  GPIO_InitStruct.Pin = SPI2_SCK_Pin|SPI2_MOSI_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
  HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */

/* USER CODE END 4 */

/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}

#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
```

```
301 #endif /* USE_FULL_ASSERT */
302
303 /************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
304
```