AUTOSAR

lecture 1

Software requirements

هوا عباره عن شرح او اقتراح لل software architecture الى هتحقق ال functionality المطلوبه من ال software دا

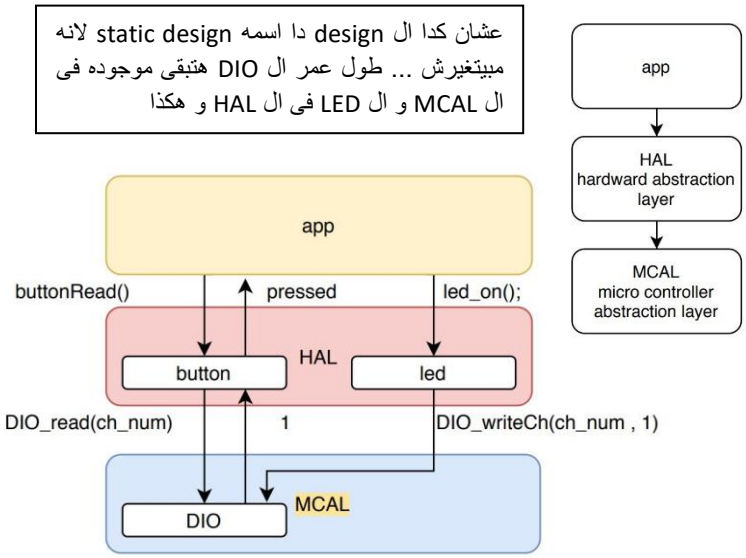
طيب هيا ايه ال requirements دى ؟

1- Functional requirements
هيا requirements خاصه بال behavior او ال tasks المطلوبه من ال SW دا
   a. Positive requirements (useful function)
   ذى ان لازم ال function دى لو دخلتلها ال input كذا هيطلع output كذا ... الى هوا الحاجه المطلوبه فعلا من ال function او ال software
   b. Negative requirements
   ان مثلا لو دخلتله input غلط هى report error او مينفذش حاجه معينه
2- Nonfunctional requirements
   a. Real time constrains
   b. Resource constrains
   ذى انى مضطر اشتغل على controller معين
3- Other requirements
   ذى انى لازم مثلا اكتب بال autosar standards

Layers

ال layers هيا جزء من تقسيمه ال code فى ال static design
يعنى كل system بيبقى فيه abstraction layer هيا layer بتعزل ال layers الى فوق عن ال layers الى تحت ذى ال windows بيعزل ال application عن ال hardware

عشان كدا ال design دا اسمه static design لانه مبيتغيرش ... طول عمر ال DIO هتبقى موجوده فى ال MCAL و ال LED فى ال HAL و هكذا



Software design

ال software design بيتقسم ل :

1- Static design
2- Dynamic design

1- Static design

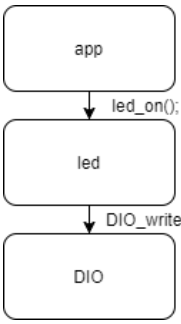
ال static design هوا انى اقسم ال software بتاعى لاجزاء كل جزء مسؤل عن function معينه كدا ال system بقى مش complex و بقى اسهل فى ال testing , debugging بدل ما ابقى ب test , debug ال system كله على بعض
افتكر :

- ال module testing هوا انى ب test كل module لوحده ... بتاع ال timer لوحده و ADC لوحده و هكذا
- ال integration testing انى اعمل test لكذا module مع بعض عشان اشوف ال interfacing بينهم و بين بعض
- ال system testing ب test ال functionality بتاعت ال system كله مع بعض
- فى test اسمه code review ... دا معناه ان حد بيص عالكود بتاعى عشان يتاكد من حاجات مينفعش يتعملها test ذى مثلا اسماء ال functions

ال static software design بيفيد فى ال reusability انى اقدر اجيب module كنت كاتبه قبل كدا و استخدمه تانى فى كل project

Testing note

لو عايز اعمل test على system ذى دا



١- هعمل test على module يعنى مثلا اجرب اعمل call ل led_on و اشوف هل فعلا هتعمل Call ل DIO_writeCh و تحط فى ال Parameters ال ch_num الصح و رقم ١ ولا لا
- ازاى هعمل module test منغير ماعمل #include DIO.h ؟؟
انى اعمل حاجه اسمها stup معناها انى اعمل Function اسمها DIO_writeCh() و اسيبها فاضيه ... لو ال LED module عملها call فعلا يبقى ال test بتاع ال led module كويس
و فى اخر ال test بنطلع test report فيه كل test و اتعمل ازاى و pass ولا fail

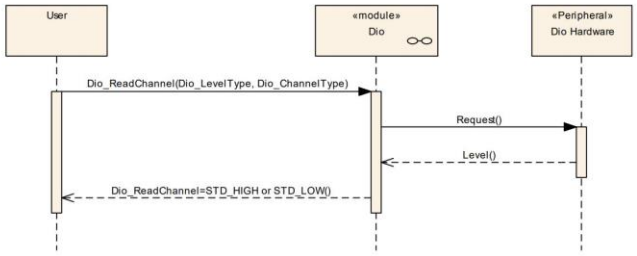
عيوب ال layer architecture

1- High over head

لان فى function calls كتيره و دا بيضيع time كبير بسبب ال context switching

2- Large Code size

Sequence diagram



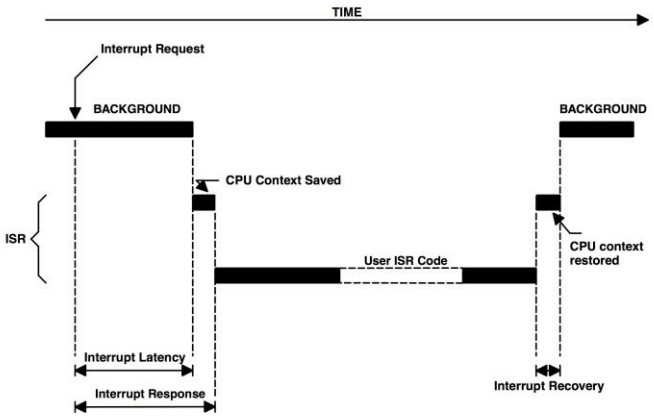
دا مثلا diagram بيوضح لو ال user عايز ي read channel بيعمل ايه ؟

بيبعت Dio_ReadChannel(Dio_LevelType, Dio_ChannelType) للDIO module و ال dio module يبعت request و يرجع بال level بتاع ال pin

ال dashed line الى راجع دا معناه ان ال function بتreturn void

https://www.autosar.org/fileadmin/user_upload/standards/classic/19-11/AUTOSAR_SWS_DIODriver.pdf (page 43)

interrupt latency , interrupt response



ال interrupt latency هوا الوقت الى بياخده ال microcontroller من ساعه ما يجى ال interrupt لغايه ما يبتدى ينفذ ال context switching

ال interrupt response الوقت الى بياخده من اول ما يحصل ال interrupt لغايه ما يبتدى تنفيذ اول سطر فى ال ISR

ليه ممكن ال interrupt يتاخر ؟

١- ممكن ابقى شغال فى ISR تانى اعلى priority (فدايما بنحاول نعمل optimization فى ال ISR) ليه ؟

a. عشان لو فى interrupt ليه priority اقل ميستناش كتير

b. عشان اى interrupt بيقطع اى task شغاله (دا فى ال RTOS) فهاكل من وقت ال task

٢- لو انا عامل disable لل interrupts عشان critical section مثلا

a. فلازم اقلل كود ال critical section عشان اقلل ال latency

و لو مضطر ازود ال critical section بضطر استخدم ال semaphores

كل ما ال latency يبقى اقل كل ما ال design يبقى احسن

2- Dynamic design

الحاجات الى بتاثر على ال dynamic design

1- Tasks
2- Priorities
3- Timing(latency ,response , jitter)
4- CPU load

هنا انا بشوف كل task هيتعملها call امتى

فكل task بتشتغل لما يحصل ال activation condition بتاعها ... ذى فى ال RTOS كدا

ذى مثلا

1- Event based task

بتستنى حد يعمل event معين

2- Interrupt based
3- Periodic task / cyclic task / time based

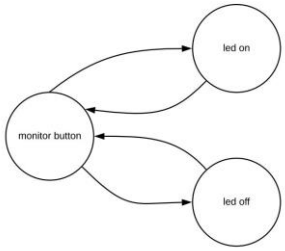
بتشتغل مثلا كل ما ال time بتاعها يجى

هنا مثلا انا الى بختار ال priority و ال tasks هيتعملها activation امتى

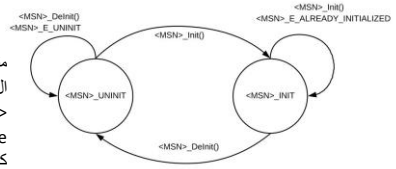
ال dynamic design ممكن نمثله بال

1- state diagram
2- sequence diagram

State diagram



دا برضو diagram بيوضح العلاقه بين ال functions و ال events و ال calls فى ال simple led / switch ل dynamic design application



مثلا ال state diagram دا بيعبر عن حاله ال module هل هوا initialized ولا لا ... <MSN> دى معناها module specific name او اسم ال module

كل module ليه global variable مثلا يائما هيبقى قيمته ب <MSN>_UNINIT> لو عملنا call لل <MSN>_DeInit()> او هتبقى قيمته <MSN>_INIT> لو عملنا call لل function الى اسمها <MSN>_Init()

CPU load

ال CPU قاعد فتره idle قد ايه يعنى مفيش tasks شغاله ... كل ما ال CPU يبقى idle اكتر كل ما ال design يبقى احسن ليه ؟

لان اكيد لو قاعد idle اى interrupt او task هتيجة هتشتغل على طول لان مفيش حاجه شغاله فبالتالى برضو ال latency , response times اقل

ال function calls الكتيره بتزود ال CPU load

لو ال task بيتعملها activation كتير (تاسك مثلا بيتعملها activation كل 1 ms) دا هيزود ال CPU load

Periodicity jitter

ان ال task بدل ما تتنفذ فى وقتها تتنفذ متاخر عشان التاسك الى قبلها قطعها interrupt و خد شويه وقت عشان ينفذ ال ISR بتاع ال interrupt دا

AUTOSAR (AUTomotive Open System ARchitecture)

دى standard عملته شركات ال automotive ... ليه ؟
عشان نقدر ن manage the complexity of systems
و عشان نسهل ال integration بين الشركات الى كلهم اصلا شغالين بال autosar standards
فادنى فى ايه ؟

1- Modularity
2- Scalability
ممكن ازود feature او module زياده منغير ما االثر على باقى ال layers
3- Transferability
انى استعمل حتت من ال application فى اكتر من ECU
جزء من ال code موجود فى ال application layer
الاجزاء فى ال application layer بيسموها component ... فى اى حته تانيه اسمها module
4- Re-usability
5- Standardized interface
كل module عنده اسامى ال functions ثابته بالتالى
اسهل فى ال integration طالما كل الشركات الى بت integrate شغاله autosar standard
6- Abstraction from HW
ال autosar بيتعامل مع ال communication بين ال hardware جوا العربيه نفسها اى حاجه برا العربيه مالوش علاقه بال standard بتاع ال autosar

MCAL micro controller abstraction layer

ال layer دى هدفها تعزل ال HW عن باقى ال layers
- الى بيعمل ال code بتاعها هما شركات tier 2 او ال hardware suppliers الى هما بيعملو ال micro controllers
- ال MCAL جواها drivers

ال module هوا algorithm اى اى software peace موجوده فى ال ECU layer
ال driverهوا اى software بي drive ال hardware
ال component هوا جزء من ال software الى موجود جوا ال application layer

MCAL : IO group

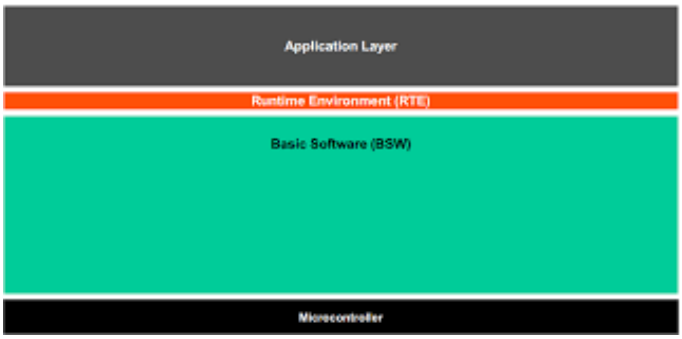
دا ال group فى ال MCAL موجود فيه اى حاجه ليها علاقه بال input , output signal ذى ال ICU , PORT , DIO

1- PORT driver

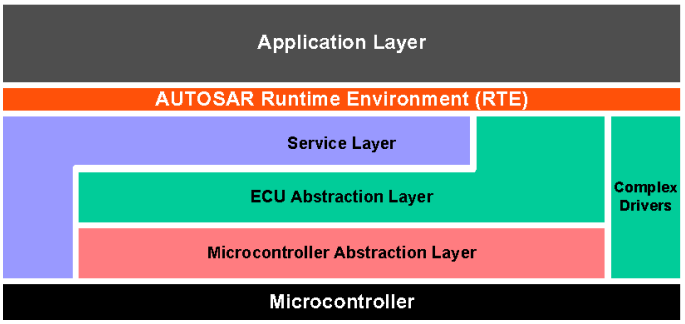
اول function بيتعملها call فى ال system هيا ال port init function ؟ ليه ؟ لانه مسؤل عن
١- بيظبط direction ال pins كلها
٢- لو output هوا الى هيطلع ال initial value بتاعتها
٣- لو input هوا الى بيظبط ال internal pullup / down resistors
٤- بيظبط ال mode بتاع ال pin ... بمعنى هل ال pin دى هتشتغل مع ال port ولا هتشتغل مع ال uart tx مثلا

AUTOSAR layers

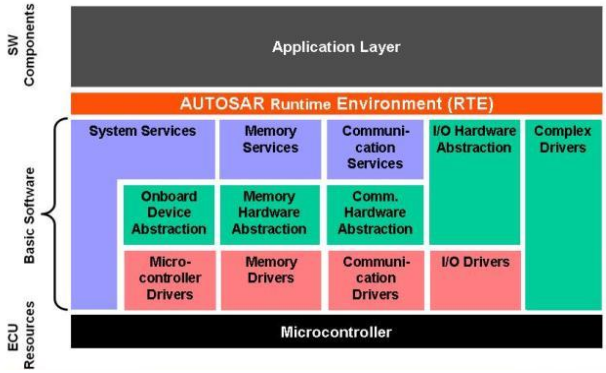
1- Basic software layer
2- RTE layer
3- Application layer



ال basic software layer نفسها بتتكون من 4 layers
ال layers بتتقسم groups فى مثلا group خاص بال IO modules و جروب خاص بال memory و هكذا
و كل مجموعه groups ليهم علاقه ببعض فى كذا layer بتتسمى stack ففى مثلا stack IO دى بتجمع كل ال IO group الى فى ال MCAL و ال service layer و ال ECU layer



1- MCAL
2- ECU abstraction layer
3- Service layer
4- Complex drivers



2- DIO driver

دا المسؤل عن انه يعمل read , write ل

- PORT
- Channel
- Group

3- ADC driver (channel)

4- PWM driver (duty , freq)

5- ICU driver (duty cycle calculation)

MCAL : memory group

دا ال group فى ال MCAL موجود فيه اى حاجه ليها علاقه بال ineternal memories

1- Internal EEPROM

ال read , write فيها بيبقى بال bytes

2- Internal flash

ال read , write بيبقى pages(4-8 byte) و ال delete بيبقى sectors(4K) الى هيا بتبقى كذا page

بستخدمها مثلا انى اسجل ال errors الى حصلت فى ال ECU

MCAL : microcontroller group

دا ال group فى ال MCAL موجود فيه اى حاجه ليها علاقه بال internal hardware الى فى ال micro controller

1- GPT general purpose timer

سواء هيشتغل one shot ولا continuous

2- WDT

3- MCU micro controller unit

دا مسؤل يظبط ال clock لكل ال peripherals الى فى ال micro controller هوا الى بيظبط ال PLL , perscaler و بيعتمد جامد على ال hardware

MCAL : communication group

دا ال group فى ال MCAL موجود فيه اى حاجه ليها علاقه بال communications سواء مع external HW على نفس ال ECU بال SPI protocol او ال communication مع اى ECU تانيه ذى ال CAN او communication مع ال sensors ذى ال LIN

1- SPI

ال SPI بيتسمى handler لانه بيقدر ي handle ال multiple requests يعنى ايه ؟

يعنى لو عندى كذا تاسك واحده عايزه تعرض على LCD و واحده عايزه تكتب فى ال external EEPROM و واحده عايزه تقرا من external ADC مثلا

لو تاسك فيهم ابتدت ال SPI هيبعت ال data لو جت ال task بتاعت ال EEPROM اشتغلت و هيا high priority هتقطعها فبالتالى ال data الى كانت بتتبعت مش هتتبعت

فكلمه handler معناها ان ال SPI عنده buffer يقدر ي handle كذا request من كذا task

2- LIN (up to 16 device , check sum error checking)

ال LIN هوا upgrade لل UART لان ال UART كان اخره Device واحد و معندوش error check sum

ال LIN هوا single master multi slaves

3- CAN

هوا بدل ال I2C هوا اه يقدر يكلم كذا device بس one device per time (device addressing) انما ال CAN هوا message addressing يعنى بيبعت ال message ب address خاص بيها و الى عايز ال message دى ياخدها غير ان ال I2C برضو معندوش error checking