CAN/TTCAN SİSTEMLERİN UPPAAL ARACI İLE MODELLENMESİ VE ZAMANLAMA DOĞRULAMASI

Çağatay ÖZDEMİR

REHİS-TTD Görev Yazılımları Müdürlüğü, ASELSAN A.Ş.
cagatayo@aselsan.com.tr

Özetçe. Bir sistemi oluşturan dağıtık elektronik kontrol birimleri arasındaki iletişim gerekleri, birimler arası veriyolunun seçimini ve tasarımını belirlemektedir. Controller Area Network (CAN) veriyolu, sağladığı bant genişliği, hata toleransı ve düşük maliyeti nedeniyle araç içi haberleşmede, akıllı evlerde ve endüstriyel kontrol sistemlerinde sıklıkla kullanılan bir protokoldür. Ancak olay tetikli haberleşme yapısı, zaman kritik mesajların sistem güvenliğini tehlikeye sokabilecek kadar gecikme yaşamasına neden olabilmektedir. CAN veriyolu protokolü üzerine geliştirilen Time Triggered CAN (TTCAN) protokolü ise zaman tetikli yapısıyla periyodik mesajların kendilerine ayrılan belirli zaman dilimlerinde gönderilebilmelerini sağlarken, aperiyodik mesajların gönderilebileceği yargılama pencerelerini de sunmaktadır. Her iki protokolde de sistemde yer alan emniyet kritik mesajların zaman kısıtları içinde gönderilebilmeleri gerekir. Bu makalede, CAN ve TTCAN veriyolu protokolü kullanan sistemlerde tasarlanan mesajlaşmalar Zamanlı Otomat (Timed Automata-TA) prensibine dayanan UPPAAL aracıyla modellenerek zamansal açıdan doğrulanacaktır. Sistem gerekleri doğrulanmış bir tasarımı gerçekleyen gömülü sistem yazılımlarının, haberleşme gecikmelerinden dolayı gerçek zamanlı emniyet kritik isterlerini karşılayamaması riskinin ortadan kaldırılması hedeflenmiştir.
Anahtar Kelimeler. CAN veriyolu, TTCAN, gerçek zamanlı sistemler, zamanlı otomat (timed automata), modelleme, biçimsel doğrulama, UPPAAL

1 Giriş

Bir gömülü sistemi oluşturan dağıtık birimler arasındaki haberleşme tipi sistemin uygulama alanına ve performans gereklerine göre belirlenir. Seçilen haberleşme tipi bant genişliği, gecikme değerleri ve bu değerlerin belirliliği (determinism), hata yakalama ve düzeltme, genişletilebilme ve yedeklenebilme gibi özellikleriyle sistem gereklerini sağlayabilmelidir [1]. Bu özelliklerin yanı sıra olay tetikli iletişime olanak sağlanması, gürbüzlük (robustness), bakım (maintainability) ve gizlilik (privacy) de gömülü sistem haberleşmelerinde aranan gereklerdendir [2]. Özellikle gerçek zamanlı sistemlerde haberleşmenin belirsiz (nondeterministic) özelliklere sahip olmaması, zaman kritik mesajların maksimum gecikme sınırları içerisinde gönderilebilmeleri büyük önem taşımaktadır. Noktadan noktaya bağlantılı haberleşmeler iki birim ara-

sında gerçek zamanlı iletişimin sağlanması konusunda başarı sağlayabilirler. Ancak, büyük sistemlerde yer alan birimlerin çokluğu yüksek miktarda kablolamaya ve haberleşme donanımının karmaşıklaşmasına neden olmaktadır. Bu durum askeri sistemlerde gözetilmeye çalışılan SWaP (Size, Weight and Power) prensibine ters düşmektedir [3]. Bu nedenlerle birimlerin ortak bir veriyolu üzerinden haberleşebilmeleri sistemlerde daha çok tercih edilmektedir. Ancak kullanılacak olan veriyolunun çoklu erişim yönteminin özellikleri ve sistemin gerçek zaman gereksinimlerine uygunluğu gözetilmelidir.

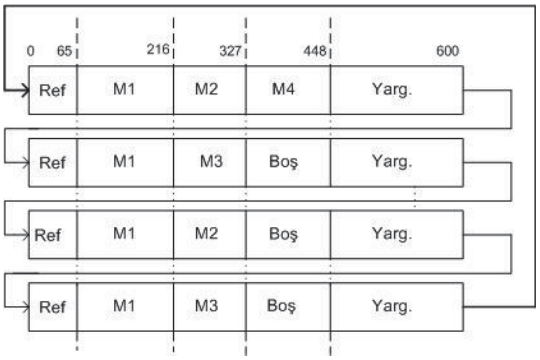
Sistemlerde önceden belirlenmiş aralıklarla zaman tetikli olarak gönderilen mesajlar periyodiktirler. Periyodik mesajların periyot zamanlarında ve kesin belirlilikle gönderilmeleri gerekir. Periyodik olmayan mesajların hepsi ise olay tetiklidir ve aperiyodik olarak adlandırılır [2]. Aperiyodik mesajların bazıları zaman kısıtına sahip değilken bunun dışında kalanlar toleranslı gerçek zaman (soft real time) ya da mutlak gerçek zaman (hard real time) özelliklerine sahip olabilir. Gömülü sistem yazılımları tasarlanırken, sistemde yer alan mesajların gerçek zamanlılığını niteleyen karakteristikleri belirlenmeli ve birimler arası haberleşmeyi sağlayan veriyolunun zaman kısıtlarını sağlayıp sağlayamayacağı doğrulanmalıdır. Bu makalede, UPPAAL aracı kullanılarak CAN/TTCAN protokolleri kullanan örnek bir sistemdeki mesajlaşmalar biçimsel olarak modellenmiş ve yaratılan benzetimlerle (simulasyon) hedeflenen zamanlamalar doğrulanmıştır. Sistem gereklerine uygunluğu doğrulanmış tasarımın gömülü yazılımlarda uygulanmasıyla görevin zamanında yapılabilmesi hedeflenmektedir.

Bu çalışmanın 2. bölümünde CAN/TTCAN protokollerinin özelliklerinden bahsedilirken, 3. bölümde bu protokolleri kullanan bir sistemin model tasarımı ve zamanlama doğrulaması aktarılacaktır. 4. bölümde ise çalışmanın sonuçları özetlenecektir.

2 CAN/TTCAN Veriyolu Protokolleri

CAN veriyolu, motorlu araçlarda, akıllı evlerde ve fabrika sistemlerinde kullanılan yaygın standartlardan birisidir. Sağladığı ortak veriyolu yapısı ve ucuzluğu sayesinde noktadan noktaya haberleşme sistemlerininin yerini almıştır. CAN veriyolu hükümetler ve savunma şirketleri tarafından desteklenen ve askeri araç teknolojilerinde kullanılmak üzere geliştirilen MilCAN protokolü gibi başka protokollere de temel oluşturarak kullanım alanlarını genişletmektedir [4]. CAN veriyolu kullanılarak elektronik kontrol birimleri arasında olay tetikli bir haberleşme ile aperiyodik mesajlar gönderilip alınabilir. Bu esnada veriyolunda yaşanan hatalar algılanarak mesajların otomatik olarak yeniden gönderilmesinin sağlanması CAN veriyolunun güvenilirliğini arttırmaktadır. CAN veriyoluna bağlı birimler fiziksel iletişim ortamını CSMA-CA (Carrier Sense Multiple Access / Collision Avoidance) yöntemine uyan öncelik tabanlı "bit yargılama" yaklaşımı sayesinde çarpışma yaşanmadan paylaşırlar. Sistemde yer alan her CAN mesajı diğer mesajlardan farklı bir öncelik değerine sahiptir. Önceliği yüksek olan CAN mesajlarına küçük ID değerleri verilmelidir. Yüksek önceliğe sahip mesajlar daha az gecikme yaşarken düşük öncelikli mesajların yaşadıkları gecikmeler daha yüksektir [5].

CAN veriyolunda tanımlanmış yüksek öncelikli mesajlar gerçek zaman kısıdını aşabilecek gecikmeler yaşayabildiği gibi periyodik mesajlar da deterministik bir periyotta gönderilemezler. Bu nedenle gerçek zaman kısıdı taşıyan mesajların zaman tetikli bir yapıyla gönderilebildiği TTCAN protokolü geliştirilmiştir [6]. Bu protokolle haberleşen birimlerin zaman senkronizasyonunu sağlayan bir zaman yöneticisi (time master) vardır [7]. Temel döngüler (basic cycle) içinde, her biri farklı amaçlar için ayrılmış zaman pencereleri bulunur. Bir temel döngü içinde yer alabilecek zaman pencereleri; referans, mesaja özel, yargılama ve boş zaman pencereleridir. Referans zaman penceresinde zaman yöneticisi sistemde yer alan diğer birimlere referans mesajı gönderir. Mesaja özel zaman pencereleri her bir temel döngüye özel olarak bir birimin tek bir mesajına ayrılmıştır; bu zaman aralığında başka bir mesaj gönderilemez. Yargılama zaman pencereleri, sistemde yer alan aperiyodik mesajlar için kullanılabilen ve CAN bit yargılaması kullanılarak mesajların gönderilebildiği zaman pencereleridir. Boş zaman pencerelerinde ise hiç bir mesaj gönderilmez, bu pencereler ileride sistem genişletilirken kullanılmak üzere saklanır.



Şekil 1. TTCAN Sistem Matrisi

TTCAN protokolü kullanılan sistemlerde yer alan tüm mesajların zaman çizelgelemesinin tek bir temel döngüde yapılması çok zor olacağı için birbirini takip eden temel döngüler düşünülerek planlama yapılmaktadır. Zamanlama çizelgesinin yapıldığı bu temel döngüler dizisine sistem matrisi denilmektedir. Şekil 1'de sistem matrisini oluşturan her bir temel döngü matrisin satırı olarak düşünülürken, temel döngülerde yer alan zaman pencereleri sistem matrisinin zaman kolonlarını oluşturmaktadır. Sistemde yer alan birimler, mesaj gönderebilmek ve alabilmek için sadece kendilerine ait zaman pencerelerinden haberdardır. TTCAN protokolünün bulunduğu sistemlerde birimler arası senkronizasyonu zaman yöneticisinin gönderdiği referans mesajı sağlamaktadır. TTCAN protokolünün 1. seviyesinde referans mesajla birlikte kaçıncı temel döngüde bulunulduğu değeri birimlere gönderilmektedir. TTCAN protokolü, tüm bu özellikleriyle zaman tetikli ve olay tetikli iletişimi birlikte sağlayan melez bir haberleşme biçimi olarak düşünülebilir.

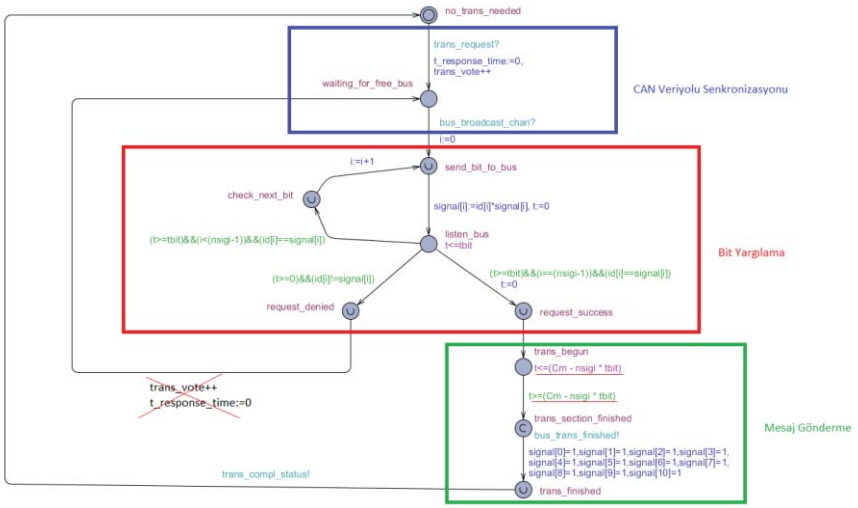
3 Sistem Modellemeleri ve Zamanlama Doğrulaması

3.1 Zamanlı Otomat ve UPPAAL Modelleme Aracı

Sınırlı durum makinelerinin (Finite State Machines) gerçek zamanlı analog saat değerleriyle genişletilmiş haline zamanlı otomat denilir [9]. Analog saat değerlerinin tam sayı değerlerle karşılaştırılması, durumlar arası geçişlerin analog saat değerleriyle korunması (guard), durumlarda geçirilebilecek zamanın lokal değişmezler yardımıyla sınırlandırılması (local invariant) gibi özellikleriyle zamanlı otomatlar gerçek zamanlı sistemlerin modellenmesinde kullanılabilirler [10]. UPPAAL aracı zamanlı otomatları kullanarak eş zamanlı görevlerin gerçek zamanlı olarak modellenmesinde ve benzetim çalışmalarıyla doğrulanmasında kullanılır [10]. Modellenen farklı otomatlar arasındaki senkronizasyon, çeşitli kanallar kullanılarak sağlanmaktadır. UPPAAL ile modellenen gerçek zamanlı sistemler, benzetim ve sorgulamalarla doğrulanabilir.

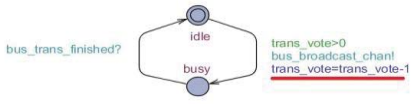
3.2 CAN Veriyolu Modellemesi ve Zamanlama Doğrulaması

CAN veriyolunun zamanlı otomat kullanılarak modellenmesi ve doğrulanması çalışmalarına [11]'de yapılanlar incelenerek ve uygulanarak başlanmıştır. [11]'de sunulan otomatlar olabildiğince basit tutulmuştur. Örneğin, zamanlı otomat olarak modellenen periyodik ve aperiyodik mesajların hepsinin farklı birimler tarafından gönderildiği varsayılarak benzetim yapılmıştır. Ayrıca CAN protokolünün sağladığı hata algılama ve düzeltme fonksiyonları da modellenmemiş, benzetim çalışmalarının hatasız bir ortamda gerçeklendiği varsayılmıştır. Ancak [11]'de verilen modeller UPPAAL aracında gerçeklendiğinde beklendiği gibi çalışmadığı görülmüştür. Bu modeller gerçeklenerek başlanan çalışmada [11]'de yer alan hatalar düzeltildikten sonra uyumlu sonuçlara ulaşılmıştır.



Şekil 2. Alıcı/Verici Zamanlı Otomatı

Alıcı/Verici ve CAN Veriyolu Zamanlı Otomatları. CAN veriyolunda yer alan her birimde veriyoluyla mesaj alışveriş arayüzünü elektriksel olarak sağlayan bir alıcı/verici (transceiver) yapısı bulunmaktadır [12]. Şekil 2'de yer alan alıcı/verici modelinde maviyle çerçevelenmiş kısım CAN veriyolunun müsaitlik durumuna göre sistemde yer alan tüm alıcı/vericiler arasındaki senkronizasyonu sağlamaktadır. Sistemde o anda mesaj göndermek isteyen birimlerin sayısı *trans_vote* global değişkeniyle tutulur. Mesaj gönderim isteğini *trans_request* kanalıyla alan alıcı/verici zamanlı otomatı, *trans_vote* global değişkeninin değerini arttırır. Veriyolu üzerinde, mesaj göndermek isteyen en az bir alıcı/vericinin bulunması Şekil 3'te yer alan veriyolu modelinin müsait (idle) durumdan meşgul (busy) duruma geçmesine neden olmaktadır. Veriyolu otomatı bu geçişi yaparken *bus_broadcast_chan* yayın kanalını (broadcast channel) kullanarak veriyolunun müsait olmasını bekleyen alıcı/vericilere durum değişikliğini bildirir. Böylelikle *waiting_for_free_bus* durumunda bulunan tüm alıcı/vericiler Şekil 2'de kırmızıyla gösterilen bit yargılama kısmına senkron bir şekilde girmiş olurlar. Bit yargılama kısmında yer alan *signal* değişkeni veriyolundaki gerçek sinyal değerini simgeleyen global bir değişkendir. Global *signal* değeriyle mantıksal "VE" işlemine giren *id* değişkenleri ise her bir alıcı/verici tarafından gönderilmek istenen CAN mesajlarının ID değerini simgeleyen yerel ve sabit değişkenlerdir. Gerçek CAN sistemlerinde olması gerektiği gibi, sistemde yer alan tüm mesajlar için bu değer birbirlerinden farklı seçilmiştir. *tbit* değeri, CAN veriyolunda 1 bitin gönderilip veriyolunun dinlenmesi için geçen süredir ve bit zamanı olarak adlandırılır. Gerçek sistemlerin doğru senkronizasyonla çalışabilmesi için *tbit* değeri birimlerin birbirlerinden uzaklığına ve çeşitli donanımsal özelliklere göre hesaplanmaktadır [13]. [11]'de bit zamanı için kesin bir değer verilmemiş, istenirse bir değer aralığı olarak alınabileceği belirtilmiştir. Yapılan araştırmalara göre 1 Mbit bant genişliğine sahip ve maksimum 40 metreye kadar haberleşme sağlayan bir CAN veriyolunda bit zamanı 1 mikrosaniye olarak alınabilmektedir [14]. Alıcı/verici otomatında yer alan *nsigi* değişkeni ise bit yargılamasına girecek olan CAN ID'sindeki bit sayısıdır. [11]'de bit sayısının ne olacağı belirtilmemiştir. CAN 2.0A standardında CAN ID'si 11 bit iken CAN2.0B'de 29 bittir. Buradaki benzetimde CAN 2.0A mesaj çerçevesinin 11 bitlik ID değerleri ve 1 mikrosaniyelik bit zamanı (*tbit*) kullanılmıştır. *sendbit_to_bus* durumunda bit yargılamasında yer alan birimlerden *id* değeri en küçük olan veriyolu üzerinden mesajını göndermeye başlar.



Şekil 3. CAN Veriyolu Zamanlı Otomat

Yapılan Düzeltmeler. [11]'de aktarılan otomatlardaki yanlış kısımlar düzeltildiğinde CAN veriyolu üzerinden gönderilen mesajların [5]'teki teorik maksimum tepki sürelerini (response time) yaşadığı görülmüştür. Buna ek olarak, sistem üzerine

yapılan mantıksal sorgulamaların da doğrulanmasıyla CAN veriyolu modelinin doğru bir şekilde çalıştığı tespit edilmiştir. [11]'de aktarılan çalışmaya yapılan düzeltmeler şunlardır:

- Şekil 2'de gösterilen alıcı/verici otomatında bit yargılaması kaybedilerek *request_denied* durumuna girildikten hemen sonra *waiting_for_free_bus* durumuna geçilmektedir. Bu geçiş esnasında yapılan *trans_vote* global değişkeninin arttırılması işlemi kaldırılmıştır. Çünkü *trans_vote* değerinin arttırılması, iletilmeyi bekleyen tüm mesajlar iletilse bile; *trans_vote* değerinin sıfırdan büyük kalmasına, veriyolu otomatının yanlış bir şekilde meşgul (busy) durumuna geçmesine ve sistem modelinin kilitlenme (deadlock) yaşamasına neden olmaktadır. Ayrıca aynı geçişte *t_response_time* değerinin sıfırlanması işlemi de kaldırılmıştır. Çünkü *t_response_time* analog saat değeri gönderilmek istenen mesajın yaşadığı tepki süresini ölçmek için tutulmaktadır ve bu noktada sıfırlanması tepki süresinin en büyük nedenlerinden biri olan bit yargılaması kayıplarının hesaba katılmaması anlamına gelmektedir. Bu durumda sistemdeki mesajların zamanlama doğrulamaları yapılamaz.
- Şekil 3'te gösterilen CAN veriyolu otomatı müsait durumdan meşgul duruma geçerken altı kırmızıyla çizili olan işlemi yaparak *trans_vote* global değişkeninin değerini bir azaltmaktadır. Ancak [11]'de verilen veriyolu otomatı aynı geçişte *trans_vote* değerini sıfıra eşitlemektedir. Eğer bu değer sıfıra eşitlenirse, bit yargılamasını kazanan mesaj gönderildikten sonra veriyolu otomatı, yeni bir alıcı/verici otomatı *trans*_vote değerini arttırmadığı sürece müsait durumda bekler. Yani sistemde gönderilmeyi bekleyen mesajlar bu süre boyunca gönderilmezler. Ancak gerçek sistemlerde CAN veriyolu müsait olur olmaz alıcı/vericiler bit yargılamasına başlamaktadırlar. Bu nedenle *trans_vote* değişkeni sıfırlanmamalı, gönderilmeyi bekleyen mesajların sayısını takip edebilmek için değeri bir azaltılmalıdır.
- [11]'de C_m değeri, bit yargılamasını kazanan alıcı/vericilerin gönderdikleri CAN mesaj çerçevesinin deterministik iletim süresi olarak tanımlanmaktadır. Ancak mesaj gönderecek olan alıcı/verici otomatı, bit yargılaması esnasında mesaj çerçevesi içinde yer alan *nsigi* adet *id* bitini *nsigi*tbit* kadar süre içinde göndermiştir. Bu noktadan sonra beklenmesi gereken süre C_m - *(nsigi * tbit)* kadar olmalıdır. [11]'de bu nokta düşünülmediği için benzetim çalışmaları sonucu bulunan maksimum tepki süreleriyle [5]'e göre hesaplanan teorik süreler birbirleriyle uyuşmamaktadır.

Zamanlama Doğrulaması. Tablo 1'deki örnek sistemde CAN protokolü kullanarak haberleşen üç birim arasındaki periyodik ve aperiyodik mesajlar ve mesajların zaman karakteristikleri görülmektedir. Aperiyodik mesajların gerçek zaman karakteristiklerinden olan minimum ara süresi (minimum inter-arrival time), aynı mesajın üst üste iki gelişi arasındaki minimum süreyi tanımlar. Aperiyodik mesajların önceliklendirilmeleri maksimum iletim sürelerine göre yapılmıştır. Maksimum iletim süresi daha az olan mesaja daha yüksek öncelikli CAN ID verilmiştir (Earliest Deadline First).

CAN ID	Mesaj Türü	Birim	Bit Uzunluğu (*bu*)	Periyot–Min. Ara Süresi(us)	Maksimum İletim Süresi(us)	**Maksimum Tepki Süresi(us)**
1	P	1	135	600	600	**240**
2	P	1	95	1200	1200	**335**
3	P	2	95	1200	1200	**430**
4	P	2	105	2400	2400	**515**
5	A	3	65	600	600	**580**
6	A	2	85	1200	1200	**645**
7	A	1	65	1200	1200	**645**

Tablo 1. CAN Mesajları Zaman Karakteristikleri ve Maksimum Tepki Süreleri
P:Periyodik, **A**: Aperiyodik

Yapılan benzetim esnasında, her mesajın birbirinden bağımsız olarak gönderilebilmek için bit yargılamasına katılabildiği varsayılmıştır. İşletim sisteminden kaynaklanan zaman seğirmesi (jitter) sıfır kabul edilmiştir.

Maksimum tepki süresi, en kötü koşullar altında bir mesajın gönderilmek istendiği zamanla veriyolu üzerinden tamamen iletildiği zaman arasındaki süre olarak tanımlanabilir. Tablo 1'de yer alan maksimum tepki süreleri, ilgili analog saat değerlerine birbirini takip eden maksimum limit sorgulamaları yapılarak koşulun sağlandığı minimum zaman değerleri olarak bulunmuştur. Sistemde yer alan bütün mesajların maksimum iletim süreleri içinde gönderilebildikleri görülmektedir. Ancak, periyodik mesajların veriyolu üzerinde iletilmeye başlaması gecikme yaşadığından istendiği gibi periyodik olarak gönderilememektedirler.

3.3 TTCAN Veriyolu Modellemesi ve Zamanlama Doğrulaması

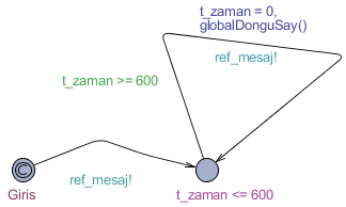
TTCAN protokolünün 2. seviyesinde zaman yöneticileri, diğer birimlere global saat değerini göndererek zaman kayması (drift) problemini çözmektedir [8]. Ancak UPPAAL aracıyla yapılan benzetimde, tüm zamanlı otomatların analog saat değerleri senkron bir şekilde artacağı için bu problem görülmeyecektir. Bu nedenle protokol 1. seviye olarak seçilmiştir.

Tablo 1'deki mesajların zaman karakteristiklerine uygun sistem matrisi Şekil 1'de görülmektedir. Şekildeki periyodik mesajlar CAN ID değerlerine uygun bir şekilde adlandırılmıştır; örneğin M1 ile gösterilen mesajın ID değeri 1'e eşittir. Sistem matrisi, her biri 600 us uzunluğunda olan ve zaman yöneticisinin gönderdiği referans mesajla başlayan dört adet temel döngüden oluşmaktadır. Periyodik mesajların hepsi periyot değerlerine uygun mesaja özel zaman pencerelerine atanmıştır. Sistem matrisinde, aperiyodik mesajların gönderilebileceği yargılama pencereleri her temel döngünün sonunda yer alır. Bunlar dışında kalan diğer pencereler ise boştur. 1. seviye TTCAN protokolündeki referans mesajı en az 1 bayt data alanına sahiptir [7]. Bu nedenle, Şekil 1'de gösterilen referans mesajı penceresi 1 bayt data alanına sahip bir CAN mesajının bit doldurma (bit stuffing) nedeniyle alabileceği maksimum uzunlu-

ğun iletim süresine eşittir. Mesaja özel zaman pencereleri içinde gönderilmesi gereken mesajların gönderilmeye başlamaları için 16 bit zamanı (*tbit=1 us*) beklenir. Eğer 16 us içinde mesajın gönderilmesine başlanmazsa mesaj gönderimi sonraki zaman pencresini etkilememesi için durdurulur. Şekil 1'de yer alan mesaja özel zaman pencerelerinin uzunluğu *(bu+16)*tbit* değerine eşit seçilmiştir [15].

Zamanlı otomatlarla yapılan sistem modellemesi hata durumlarını içermemektedir. Modellemenin tamamen hatasız bir ortam için yapıldığı varsayılmıştır. Birimler tarafından gönderilen mesajlar diğer tüm birimler tarafından başarıyla alınabilmektedir. Sistemde işletim sisteminden kaynaklı zaman seğirmesi sıfır kabul edilmiştir.

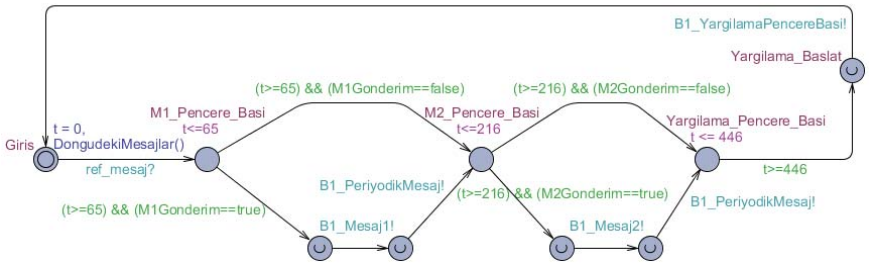
Zaman Yöneticisi. TTCAN protokolü birden fazla zaman yöneticinin yer alabileceği bir yapıya sahiptir. Ancak hatasız ortam varsayımı nedeniyle ve benzetimleri daha basit tutabilmek amacıyla sadece bir adet zaman yöneticisi otomatı yaratılmıştır. Şekil 4'te gösterilen zaman yöneticisi otomatı *t_zaman=0* anında sistemde yer alan birimlere *ref_mesaj* yayın kanalını kullanarak referans mesaj gönderir, böylelikle ilk temel döngüyü (*globalDonguSayisi=0*) başlatmış olur. Zaman yöneticisi *t_zaman=600* anında yeni bir temel döngünün başladığını diğer birimlere bildirirken, *globalDonguSayisi* değişkenini bir arttırarak temel döngüleri de sayar. UPPAAL aracında tanımlı "int" tamsayı değişken tipi [-32768, 32767] aralığında yer alabildiği için, tamsayı taşmasını (integer overflow) önlemek amacıyla *globalDonguSayisi* değişkeni 30000 değerine ulaştığında sıfırlanır.



Şekil 4. Zaman Yöneticisi Otomatı

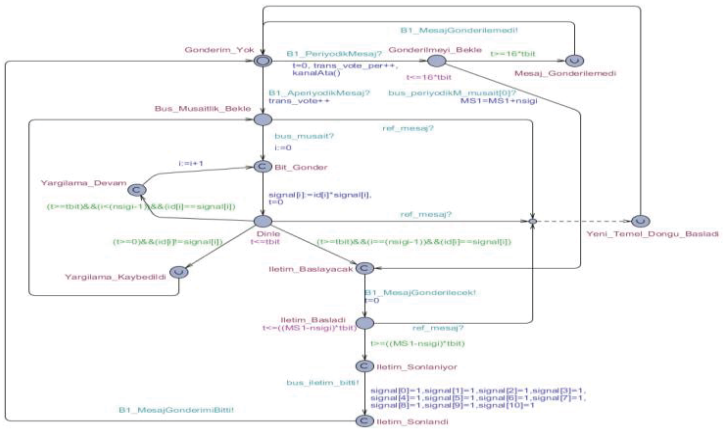
Birim Planlayıcı. TTCAN protokolünde, her birim kendi mesajlarının sistem matrisi içinde hangi zamanda planlandığı bilgisini tutmaktadır. Periyodik bir mesaja sistem matrisi içinde atanmış olan zaman pencereleri, o mesaja özel döngü ofseti (cycle offset) ve tekrar faktörü (repeat factor) parametreleriyle belirlenir. Şekil 1'deki sistem matrisinde görülen M1 için döngü ofseti 0, tekrar faktörü 1 iken; M3 için döngü ofseti 1 ve tekrar faktörü 2'dir. Birim planlayıcılar, bu parametreleri ve her referans mesajıyla güncellenen *globalDonguSayisi* nı kullanarak o temel döngüde hangi mesajın gönderileceğine karar verirler. Şekil 5'te görülen birim planlayıcı, referans mesajını her aldığında *DongudekiMesajlar* fonksiyonunu çağırarak *M1Gonderim* ve *M2Gonderim* boolean değerlerine atama yapar. Böylelikle periyodik mesajların, bir sistem matrisi içinde planlandığı temel döngüde gönderilmeleri sağlanır. Bir temel döngü içinde gönderilmesi beklenen periyodik mesajların doğru zamanda gönderilmeleri durumlara atanmış lokal değişmezler ve geçişlere

konulan zaman korumaları yardımıyla sağlanır. Örneğin, Şekil 1'de planlanan M1'e ait zaman penceresi döngü zamanı 65'e eşitken başladığı için Şekil 5'te *M1_Pencere_Basi* durumundan geçişin yapılması için tam olarak otomatın analog saat değerinin 65 olması tasarlanmıştır.



Şekil 5. Birim-1 Planlayıcı Otomatı

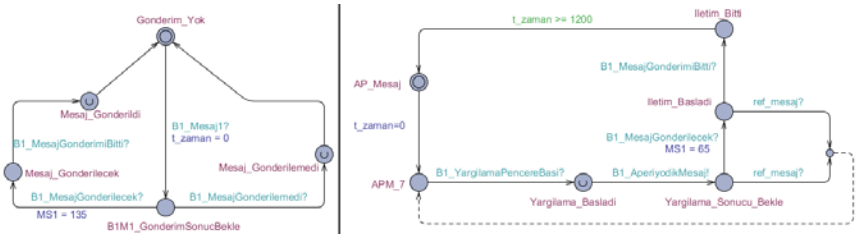
Birim Alıcı/Verici. Şekil 6'da Birim 1'e ait periyodik ve aperiyodik mesajların gönderilmesini yöneten bir alıcı/verici otomatı görülmektedir. *Gonderim_Yok* durumundayken planlayıcı otomatından *B1_PeriyodikMesaj* kanal mesajını alınca



Şekil 6. Birim-1 Alıcı/Verici Otomatı

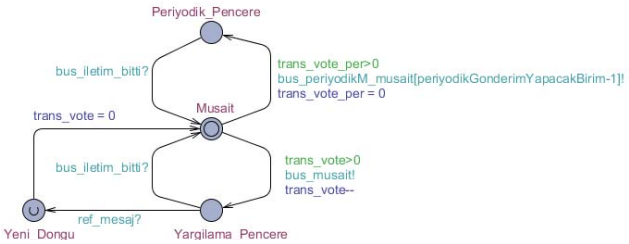
Gönderilmeyi_Bekle durumunda, mesajın gönderilmesinin başlayabilmesi için 16 bit zamanı kadar bekler. Bu süre içinde veriyolu otomatının müsait olduğunu *bus_periyodikM_musait* kanalıyla alırsa mesaj gönderilmeye başlar. Eğer 16 bit zamanı içinde veriyolu müsait değilse, periyodik mesaj gönderilmez. Aperiyodik mesajlar gönderilmeden önce ise bit yargılaması yapılır. Bit yargılaması ya da mesaj gönderimi esnasında zaman yöneticiden *ref_mesaj* alınırsa *Yeni_Temel_Dongu_Basladi* durumuna geçilerek mesaj gönderimi durdurulur. Böylece sistem matrisinin planlandığı gibi devam etmesi sağlanır.

Periyodik ve Aperiyodik Mesajlar. Periyodik ve aperiyodik mesajların gönderim durumunu takip edebilmek ve ilgili zaman ölçümlerini alabilmek için Şekil 7'de görülen otomatlar her mesaj için modellenmiştir. Periyodik mesajlar, planlayıcılar tarafından kendilerine özel kanallar sayesinde *Gönderim_Yok* durumundan çıkıp gönderim sonucunu beklemeye başlarlar. Bu durum geçişinde lokal saatleri olan *t_zaman* sıfırlanarak ölçümlerin doğru alınması sağlanır. Aperiyodik mesajlar ise bit yargılaması penceresinin geldiğini planlayıcıların gönderdiği yayın kanalı mesajlarıyla anlayıp yargılama sonucunu beklemeye başlarlar. Aperiyodik mesajlar gönderilmeden yeni bir temel döngü başlarsa, bir sonraki yargılama penceresini beklerler. Aynı aperiyodik mesajın birbirini takip eden iki gönderim isteği arasında en az minimum ara süresi kadar zaman olması sağlanır (*t_zaman>=1200*). Sistemdeki bütün mesajlar, alıcı/vericilerinin göndereceği bit sayısını kendi bit sayılarına eşitler *(MS1=135)*.



Şekil 7. Periyodik ve Aperiyodik Mesaj Otomatları

Veriyolu. Şekil 8'deki veriyolu otomatının müsaitlik durumuna göre sistemde yer alan bütün alıcı/vericiler periyodik ve aperiyodik mesajlarının gönderimini düzenler. Aperiyodik bir mesajın gönderimi esnasında eğer yeni bir temel döngü başlayacak olursa, aperiyodik mesaj gönderimi durdurulduğu için veriyolu da müsait durumuna geçer.



Şekil 8. Veriyolu Otomatı

Zamanlama Doğrulaması. Tablo 1'e uygun otomatlar yaratılarak yapılan benzetim çalışmaları esnasında sistemin doğru çalıştığına emin olmak için mantıksal sorgulamalar yapılmıştır. Kilitlenme (deadlock) sorgulaması sistemin mantıksal olarak çalışamayacak bir duruma girip girmediğini anlamak için yapılmış ve kilitlenmenin gerçekleşmediği görülmüştür. Ayrıca, alıcı/verici otomatlarının aynı anda *Iletim_Basladi*

durumlarında bulunup bulunmadığı sorgulanmış ve her alıcı/vericinin farklı zamanlarda iletim yaptığı görülmüştür. Bu mantıksal sorgulamalarla benzetimin doğru çalıştığı sonucuna varılmıştır. Sonrasında her bir mesaj için maksimum tepki süreleri ölçülmüştür. Elde edilen ölçüm değerleri Tablo 2'de görülmektedir.

CAN ID	Mesaj Türü	Maksimum İletim Süresi(us)	**Maksimum Tepki Süresi(us)**
1	P	600	**135**
2	P	1200	**95**
3	P	1200	**95**
4	P	2400	**105**
5	A	600	**598**
6	A	1200	**1198**
7	A	1200	**1178**

Tablo 2. TTCAN Benzetimi Maksimum Tepki Süreleri

Tablo 2'de görüldüğü üzere kendilerine ait zaman pencerelerinde gönderilen periyodik mesajlar hiç gecikme yaşamadıkları için veriyolunun iletim hızında gönderilmişlerdir. Ancak aperiyodik mesajlar bit yargılamasındaki önceliklerine göre gönderildikleri için belli bir gecikme yaşamaktadırlar. Bu gecikmelere rağmen mesajların maksimum tepki süreleri maksimum iletim sürelerinden küçük çıktığı için sistem matrisinin zaman kriterlerine uygun olduğu doğrulanmıştır.

4 Sonuçlar

[11]'de aktarılanların UPPAAL aracıyla gerçeklenmesi ve düzeltmeler yapılmasıyla başlanan çalışmaya örnek bir sistemin gerçek zaman kıstaslarına uygunluğu sorgulanarak devam edilmiştir. CAN veriyolu ile tüm mesajlar maksimum iletim sürelerinde iletilebilmektedir; ancak periyodik mesajlar gecikme yaşamaktadır. Aynı örnek sistemin, TTCAN protokolü kullanması durumunda sahip olacağı zamanlama özelliklerini incelemek için uygun bir modelleme yapılmıştır. Yapılan benzetimlerde periyodik mesajların hiçbir gecikme yaşamadıkları ve her mesajın maksimum iletim süresi içinde gönderilebildiği görülmüştür. Bu karşılaştırmayla, sistemdeki mesajların zaman karakteristiklerine TTCAN protokolünün daha uygun olacağı sonucu çıkarılabilir.

Sistemde yer alan mesajların zaman karakteristiklerine uygun bir şekilde gönderilip gönderilemeyecekleri, matematiksel olarak da incelenebilir. Ancak, zamanlı otomatlar kullanılarak yapılan benzetimlerle gereksinim analizi aşamasında sistemdeki birim sayısı, birimlerin göndereceği mesajlar, mesajların zaman karakteristikleri ve sistem matrisinin tasarımı daha net gözlemlenebilmektedir.

Çalışmalar boyunca zamanlama doğrulamasının UPPAAL kullanılarak yapılması sırasında sorgulama sonuçlarının alınması için beklenen zaman ve kullanılan bellek fazla olduğu için özellikle TTCAN modellemesi basit tutulmaya çalışılmıştır. Örneğin, birimlerin alıcı/verici otomatlarında veriyolu müsait olduktan sonra periyodik

mesajlar *Gonderilmeyi_Bekle* durumundan *Bit_Gonder* durumuna geçerek kendi kendilerine bit yargılaması yapabilirlerdi. Ancak benzetimde fazla zaman harcamamak için bit yargılaması yapılmadan direkt geçiş yapılmış, doğru mesaj zamanlamasını sağlamak içinse mesajın bit sayısına ID bit sayısı kadar ekleme yapılmıştır. Bu tip kısa yollar ve UPPAAL aracının sunduğu optimizasyonlarla çalışılmıştır.

Gömülü sistemlerde gereksinimlerin yapılabilirlik analizleri çok önemlidir. Aksi takdirde, geliştirme aşamasında yaşanabilecek sorunlar gereksinim ve tasarım değişikliklerine neden olabilmektedir. Zamanlı otomatlar ve UPPAAL kullanılarak, CAN ve TTCAN sistemler modellenip zamansal olarak doğrulanabilir. Doğrulanmış bu tasarımların sitemde uygulanması riskleri azaltacaktır.

5 Teşekkür

Yazar, desteklerinden dolayı ODTÜ Elektrik Elektronik Mühendisliği'nde öğretim üyesi olan Doç. Dr. Cüneyt F. BAZLAMAÇCI'ya, yönlendirmelerinden dolayı çalışma arkadaşı olan Mustafa DURSUN'a, araştırmalar esnasında istatistiksel yöntemlere ilişkin yardımlarından dolayı Dizem SİVASLIGİL'e, akademik araştırmalara desteğinden dolayı ASELSAN A.Ş.'ye teşekkürlerini sunar.

6 Kaynakça

1. Upender, B.P., Koopman, P.J.: Communication Protocols for Embedded Systems.
2. Marwedel P. Embedded System Design Embedded Systems Foundations of Cyber-Physical Systems, 2nd edn.
3. Size, Weight and Power (SWaP), http://www.cwcdefense.com/technology/size-power.html
4. MILCAN Protokolü anasayfası, http://www.milcan.org/index.html
5. Tindell, K., Burns A., Wellings A.J.: Calculating CAN Message Response Times
6. Führer, T., Müller, B., Dieterle, W., Hartwich, F., Hugel, R., Walther, M. : Time Triggered Communication on CAN (Time Triggered CAN-TTCAN)
7. ISO 11898-4:2004: Road vehicles — Controller area network (CAN) — Part 4: Time-triggered communication, the International Organization for Standardization
8. Hartwich, F., Müller, B., Führer, T., Hugel, R. : Timing in the TTCAN Network
9. Alur, R., Dill, D.: Automata for modeling real-time systems. In: Proceedings of 17th International Colloquium on Automata, Languages and Programming, pp. 322–335 (1990)
10. Bengtsson, J., Yi, W.:Timed Automata: Semantics, Algorithms and Tools, UNU-IIST Report No. 316 (2004)
11. Krakora, J., Hanzalek, Z.: Timed Automata Approach to CAN Verification (2004)
12. Di Natale, M.: Understanding and using the Controller Area Network (2008)
13. Hartwich, F., Bassemir, A., Robert Bosch GmbH: The Configuration of the CAN Bit Timing. In: 6th International CAN Conference
14. Kinnaird, C.: Signaling Rate Versus Cable Lenght: the CAN-bus Timing Trade-off. In: http://www.eetimes.com/documents.asp?doc_id=1274178
15. Schmidt, K., Schmidt, S.E Systematic Message Schedule Construction for Time-Triggered CAN (2007)